REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
25-06-2012	Final Report	April 2010 - March 2012

4. TITLE AND SUBTITLE

High Order Space-time Discontinuous Galerkin Cell Vertex Scheme toward Compressible Navier Stokes Equations

5a. CONTRACT NUMBER

5b. GRANT NUMBER

FA9550-10-1-0045

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Shuangzhang Tu

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Jackson State University
1400 Lynch St
Jackson, Mississippi 30217

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Office of Scientific Research
Computational Mathematics Program
875 North Randolph St.
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)

AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution A: publicly available

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This project continued our previous AFOSR project in the development of a high-order Riemann-solver-free space-time method named as the discontinuous Galerkin cell vertex scheme (DG-CVS). During the two-year project period, the method was first verified to be able to accurately solve diffusion equations as well as the advection equations without any special care as required by other semi-discrete DG methods. Numerical experiments reveal that the method is L2 optimal for both advection and diffusion equations when the basis polynomials are of odd degrees. For even-degree polynomials, the convergence order is optimal for advection equations but sub-optimal for diffusion equations. The curved boundary is approximated by high-order geometric polynomials which are consistent with the degree of the solution polynomials. The method has been tested to solve moving-mesh problems without the concern of the geometric conservation law thanks to its space-time nature. The method has also been extended to solve the shallow water equations and the level set equation which further confirms that the DG-CVS method is a truly working Riemann-solver-free method for arbitrary conservation laws without the need of eigen information of the system.

15. SUBJECT TERMS

Conservation laws, space time method, discontinuous Galerkin cell vertex scheme (DG-CVS), high order method, Riemann solver free method

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Shuangzhang Tu
U	U	U	UU	39	19b. TELEPHONE NUMBER *(Include area code)* (601) 979-1275

High Order Space-time Discontinuous Galerkin Cell Vertex Scheme toward Compressible Navier Stokes Equations

Final Report to AFOSR
Grant Number: FA9550-10-1-0045
Reporting Period: April 1, 2010 - March 31, 2012

PI: Shuangzhang Tu

Jackson State University
Jackson, Mississippi, 39217, USA

This final report summarizes our major accomplishments on the research under AFOSR Grant FA9550-10-1- 0045 during the project period between April 1, 2010 - March 31, 2012.

1 Background

This project continues our previous AFOSR project (Grant No. FA9550-08-1-0122) to extend and verify our high order space-time cell vertex scheme (DG-CVS) toward solving the compressible Navier-Stokes equation.

The DG-CVS method integrates the best features of the space-time Conservation Element/Solution Element (CE/SE) method [1] and the discontinuous Galerkin (DG) method [2]. The core idea is to construct a staggered space-time mesh through alternate cell-centered CEs and vertex-centered CEs (cf. Fig. 1 (right)) within each time step. Inside each SE (cf. Fig. 1 (left)), the solution is approximated using high-order space-time DG basis polynomials. The space-time flux conservation is enforced inside each CE using the DG discretization. The solution is updated successively at the cell level and at the vertex level within each physical time step. For this reason and the method's DG ingredient, the method was named as the space-time discontinuous Galerkin cell-vertex scheme (DG-CVS).

DG-CVS equally works on higher dimensions on arbitrary grids. Figure 2 shows the conservation elements and solution elements on quadrilateral meshes and triangular meshes. Obviously, the definitions of CEs and SEs on higher dimensions are analogous to that for 1-D meshes (cf. Fig. 1). Figure 3 demonstrates the

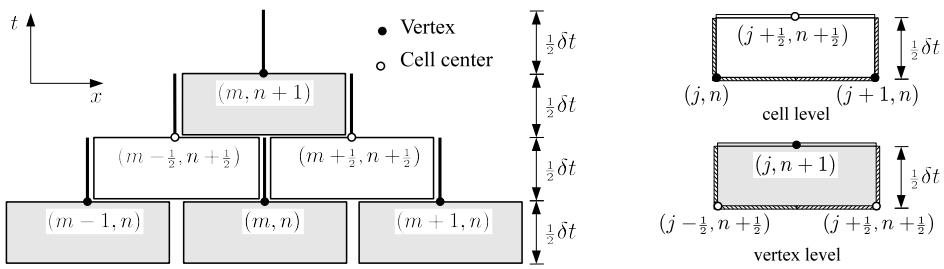


Figure 1: Solution elements (SEs) and conservation elements (CEs) in the $x-t$ domain. Left: solution elements and right: conservation elements.

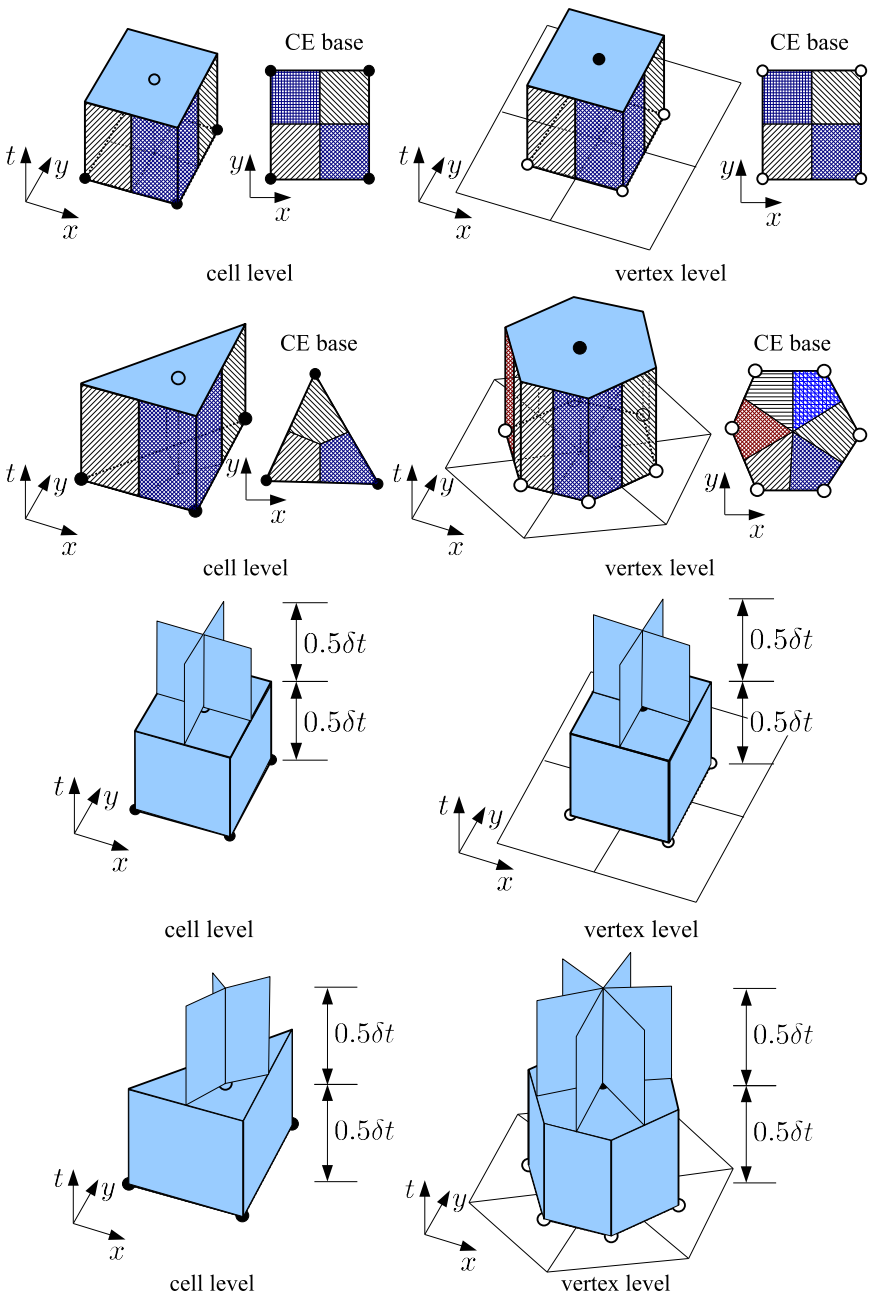


Figure 2: Conservation elements (CEs) and solution elements (SEs) in the $x-y-t$ domain. First row: CEs for rectangular meshes, second row: CEs for triangular meshes, third row: SEs for rectangular meshes, and fourth row: SEs for triangular meshes.

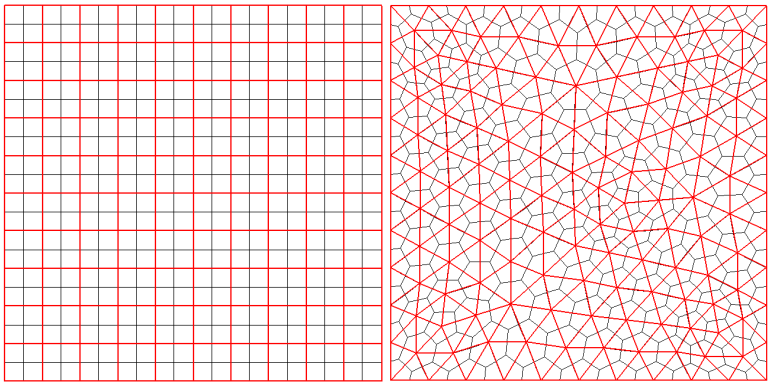


Figure 3: Dual meshes for the solution updating at the cell level (in red) and the vertex level (in black). Left: rectangular mesh and right: triangular mesh.

resulting dual mesh at the cell level and the vertex level for both rectangular meshes and triangular meshes, respectively.

The main features of DG-CVS are summarized as follows:

- *based on space-time formulation.*
- *arbitrary high-order accuracy in both space and time.* Space and time are handled in a unified way based on space-time flux conservation and high-order space-time discontinuous basis functions. This is in contrast to semi-discrete methods where the temporal order of accuracy is limited by the Backward Difference Formula (BDF) or the multi-step Runge-Kutta method.
- *Riemann-solver free.* DG-CVS does not need a (approximate) Riemann solver to provide numerical fluxes as needed in finite volume or traditional DG methods. The Riemann-solver-free feature offers two-fold advantages. First, this Riemann-solver-free approach eliminates some pathological behaviors (e.g., carbuncle phenomenon, expansion shocks, etc.) associated with some Riemann solvers. Second, it is suitable for any hyperbolic PDE systems whose eigenstructures are not explicitly known.
- *reconstruction free.* DG-CVS solves for the solution and its all spatial and temporal derivatives simultaneously at each space-time node, thus eliminating the need of reconstruction.
- *suitable for arbitrary spatial meshes.* The CE and SE definitions in DG-CVS are independent of the underlying spatial mesh and the same definitions can be easily extended from 1-D to higher-dimensions (cf. Fig. 2) without any ambiguity. Though not shown, the same CE and SE definitions also apply to meshes with hanging nodes.
- *highly compact regardless of order of accuracy.* Only information at the immediate neighboring nodes will be needed to update the solution at the new time level. Compactness eases the parallelization of the flow solver.

In the previous project, the DG-CVS method has been developed for solving hyperbolic conservation laws including the scalar advection equation and the compressible Euler equations.

2 Summary of Major Accomplishments

During the last two years, we have accomplished the following:

- *Verified that the DG-CVS method solves the time-dependent diffusion equations as well as the advection equations without any special care on diffusion terms.* Thanks to the staggered space-time conservation elements, both the advective and diffusive flux are continuous and unique across the spatial cell interface. Therefore, no extra reconstruction or recovery or ad hoc penalty and coupling terms are needed to avoid the "variational crime" and ensure the consistency of the variational form for diffusive terms. For this reason, DG-CVS is conceptually simper than other existing DG methods for diffusion equations. This paves the way to solve compressible Navier-Stokes equations.

- *Conducted the convergence rate for diffusion equations.* The convergence rate is L_2 optimal when the degree of basis polynomials is odd and sub-optimal when the degree of basis polynomials is even. In other words, DG-CVS is $(p+1)$th order accurate for the solution and pth order accurate for the solution gradients for odd p. When p is even, the convergence rate of DG-CVS is still optimal for advection equations but sub-optimal for diffusion equations. In practice, one can choose the odd p for best performance when solving advection-diffusion equations.
- *Extended to more practical problems involving curved boundaries.* The curved boundary is approximated by high-order geometric polynomials which is consistent with the high-order solution polynomials. This is indispensable in high-order methods to avoid spurious entropy generation on the curved boundary.
- *Extended to solve the shallow water equations.* The DG-CVS method provides an alternative Riemann-solver free approach for shallow water equations. Being able to solve both Euler equations and shallow water equations confirms that DG-CVS is truly a working Riemann-solver-free approach which can be used to solve arbitrary conservation laws where eigenstructures are difficult to obtain (e.g., magneto-hydrodynamics equations) in the future.
- *Extended to solve the level set equation.* The DG-CVS method is able to resolve the interface evolution governed by the level set equation.
- *Extended to solve moving-mesh problems.* Thanks to the space-time nature of the DG-CVS method, the geometric conservation law involving moving meshes is automatically satisfied. Therefore, the DG-CVS method has great potentials in moving boundary problems (e.g., fluid-structure interaction).

3 Description of the Space-Time Discontinuous Galerkin Cell-Vertex Scheme

The high-order space-time DG-CVS method is developed to solve the equations of the following type:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} - \frac{\partial \mathbf{f}_v}{\partial x} - \frac{\partial \mathbf{g}_v}{\partial y} = 0 \tag{1}$$

where $\mathbf{u}$ is the conservative state vector, $\mathbf{f}$ and $\mathbf{g}$ are advective flux vectors, and $\mathbf{f}_v$ and $\mathbf{g}_v$ are the diffusive flux vectors.

3.1 Space-Time Discontinuous Galerkin Formulation

Following the idea of the discontinuous Galerkin (DG) method, an approximate solution $\mathbf{U}^h$ is sought within each space-time solution element (SE), denoted as K. When restricted to the SE, $\mathbf{U}^h$ belongs to the finite dimensional space $\mathcal{U}(K)$ such that

Assume all the components of the conservative variables inside each SE are approximated by polynomials of the same degree, i.e

$$\mathbf{u}^h(\mathbf{x},t) = \sum_{j=1}^{N} \mathbf{s}_j \phi_j(\mathbf{x},t) \tag{2}$$

where $\{\phi_j(\mathbf{x},t)\}_{j=1}^N$ are some type of space-time polynomial basis functions defined within the solution element, $\{\mathbf{s}_j\}_{j=1}^N$ are the unknowns to be determined and N is the number of basis functions depending on the degree of the polynomial function.

The first spatial derivative of the solution can be expressed in terms of the basis functions as follows.

$$\frac{\partial \mathbf{u}^h(\mathbf{x},t)}{\partial x} = \sum_{j=1}^{N} \frac{\partial \phi_j}{\partial x}\mathbf{s}_j, \text{ and } \frac{\partial \mathbf{u}^h(\mathbf{x},t)}{\partial y} = \sum_{j=1}^{N} \frac{\partial \phi_j}{\partial y}\mathbf{s}_j \tag{3}$$

Currently, the polynomials based on the Taylor expansions are used as the basis functions. Note that, for the Taylor polynomials, the derivatives of the polynomial are a subset of the original Taylor polynomials. This allows efficient implementation when evaluating integrals involving products of the Taylor polynomials. For example, the quadratic 2-D basis functions are tabulated in Table 1.

Table 1: Derivatives of quadratic 2-D basis functions.

i	unknowns (s_i)	ϕ_i	$\frac{\partial \phi_i}{\partial x}$	$\frac{\partial \phi_i}{\partial y}$	$\frac{\partial \phi_i}{\partial t}$
1	u	1	0	0	0
2	u_x	Δx	1	0	0
3	u_y	Δy	0	1	0
4	u_{xx}	$\frac{1}{2}\Delta x^2$	Δx	0	0
5	u_{xy}	$\Delta x \Delta y$	Δy	Δx	0
6	u_{yy}	$\frac{1}{2}\Delta y^2$	0	Δy	0
7	u_t	Δt	0	0	1
8	u_{xt}	$\Delta x \Delta t$	Δt	0	Δx
9	u_{yt}	$\Delta y \Delta t$	0	Δt	Δy
10	u_{tt}	$\frac{1}{2}\Delta t^2$	0	0	Δt

Following the Galerkin orthogonality principle, multiply (1) with each of the basis functions ϕ_i ($i = 1, 2, \cdots, N$) and integrate over a space-time CE to obtain the weak form

$$\int_{\Omega_K} \phi_i \left(\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} - \frac{\partial \mathbf{f}_v}{\partial x} - \frac{\partial \mathbf{g}_v}{\partial y} \right) \mathrm{d}\Omega = 0, \quad i = 1,\, 2,\, \cdots,\, N \tag{4}$$

where Ω_K is the space-time conservation element (CE) associated with the solution element K. Note that the conservation element is identical to the solution element except for the volumeless vertical spike in the solution element. The space-time flux conservation in weak form as in (4) is for each individual space-time conservation element. Therefore, the current method can be considered as a space-time discontinuous Galerkin method.

Integrating (4) by parts results in

$$\int_{\Omega_K} \left[\frac{\partial \phi_i}{\partial t} \mathbf{u}^h + \frac{\partial \phi_i}{\partial x} (\mathbf{f}^h - \mathbf{f}_v^h) + \frac{\partial \phi_i}{\partial y} (\mathbf{g}^h - \mathbf{g}_v^h) \right] \mathrm{d}\Omega = \int_{\Gamma} \phi_i \mathbf{H} \cdot \mathbf{n} \mathrm{d}\Gamma \tag{5}$$

where $\mathbf{H} = (\mathbf{f}^h - \mathbf{f}_v^h,\, \mathbf{g}^h - \mathbf{g}_v^h,\, \mathbf{u}^h)$ is the space-time flux tensor and $\mathbf{n} = (n_x,\, n_y,\, n_t)$ is the outward unit normal of the CE boundary, i.e. $\Gamma = \partial \Omega_K$, of the space-time conservation element (CE) under consideration. Note that the partial integration is also performed on the time-dependent term, which is a salient difference between space-time DG methods and semi-discrete DG methods. As can be seen, the formulation in (5) contains both the volume integral and the surface integral.

3.2 Cell-Vertex Solution Updating Strategy

The DG-CVS inherits the core idea of the CE/SE method using a staggered space-time mesh to enforce the space-time flux conservation. However, the construction of the staggered space-time slabs in DG-CVS deviates from the CE/SE method. In DG-CVS, unknowns are stored at both vertices and cell centroids of the spatial mesh, and the solutions at vertices and cell centroids are updated at different time levels within each time step. At the beginning of each physical time step, the solution is assumed known at the vertices of the mesh, either given as the initial condition or obtained from the previous time step. Inside each new time step, the solution is updated in two successive steps. The first step updates the solution at cell centroids at the half-time level ($t^{n+1/2}$) based on the known vertex solutions at the previous time level (t^n). The second step updates the solution at vertices at the new time level (t^{n+1}) based on the known cell solutions at the previous half-time level ($t^{n+1/2}$). The same process is repeated for new time steps.

The solution updating at the cell level or the vertex level is based on the key equation (5). First divide the conservation element into the following portions:

- the interior volume Ω_K where the solution is associated with the space-time node at the new time level.
- the top surface Γ_{top} where the solution is also associated with the space-time node at the new time level.

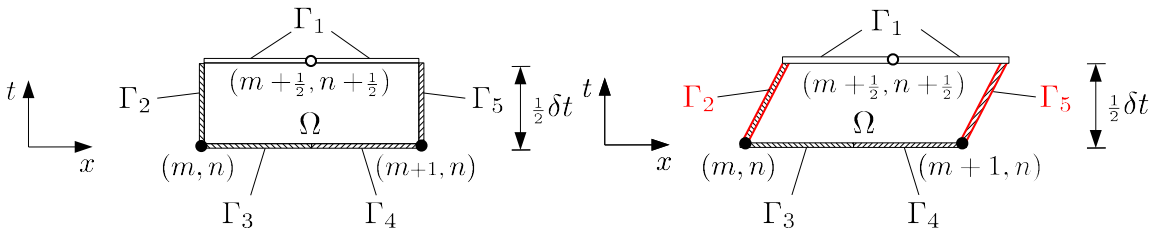


Figure 4: Illustration of space-time flux conservation on a 1-D cell-level CE. Left: stationary mesh and right: moving mesh.

- the side surfaces Γ_{side} where the solution is associated with the space-time node at the previous time level.
- the bottom surfaces Γ_{bott} where the solution is also associated with the space-time node at the previous time level.

Correspondingly, Eq. (5) can be rearranged to yield

$$\int_{\Omega_K} \left[\frac{\partial \phi_i}{\partial t} \mathbf{u}^h + \frac{\partial \phi_i}{\partial x} (\mathbf{f}^h - \mathbf{f}_v^h) + \frac{\partial \phi_i}{\partial y} (\mathbf{g}^h - \mathbf{g}_v^h) \right] \mathrm{d}\Omega - \int_{\Gamma_{\text{top}}} \phi_i \mathbf{H}^h \cdot \mathbf{n} \mathrm{d}\Gamma \qquad i = 1, 2, \cdots, N$$
$$= \int_{\Gamma_{\text{side}}} \phi_i \mathbf{H}^h \cdot \mathbf{n} \mathrm{d}\Gamma + \int_{\Gamma_{\text{bott}}} \phi_i \mathbf{H}^h \cdot \mathbf{n} \mathrm{d}\Gamma \tag{6}$$

where the left hand side contains the unknowns and the right hand side contains the known information.

Since the top and the bottom surface of the CE are horizontal, which leads to $\mathbf{H}^h \cdot \mathbf{n} = \mathbf{u}^h$ on the top face and $\mathbf{H}^h \cdot \mathbf{n} = -\mathbf{u}^h$ on the bottom face, Eq. (6) can be simplified to

$$\int_{\Omega_K} \left[\frac{\partial \phi_i}{\partial t} \mathbf{u}^h + \frac{\partial \phi_i}{\partial x} (\mathbf{f}^h - \mathbf{f}_v^h) + \frac{\partial \phi_i}{\partial y} (\mathbf{g}^h - \mathbf{g}_v^h) \right] \mathrm{d}\Omega - \int_{\Gamma_{\text{top}}} \phi_i \mathbf{u}^h \mathrm{d}\Gamma \qquad i = 1, 2, \cdots, N$$
$$= \int_{\Gamma_{\text{side}}} \phi_i \mathbf{H}^h \cdot \mathbf{n} \mathrm{d}\Gamma - \int_{\Gamma_{\text{bott}}} \phi_i \mathbf{u}^h \mathrm{d}\Gamma \tag{7}$$

To further illustrate the idea of enforcing the space-time flux conservation, consider the 1-D case shown in Fig. 4. Suppose the solution at the spacetime node $(m + \frac{1}{2}, n + \frac{1}{2})$ is to be determined. Here, m and n represents the spatial and the temporal locations of the space-time node, respectively. The boundaries of the CE associated with the spacetime node $(m + \frac{1}{2}, n + \frac{1}{2})$ is divided into five sections Γ_1, Γ_2, Γ_3, Γ_4 and Γ_5, as shown in Figure 4. Among these sections, Γ_1 belongs to the SE associated with node $(m + \frac{1}{2}, n + \frac{1}{2})$ whose solutions are to be determined, Γ_2 and Γ_3 the SE associated with node (m, n) and Γ_4 and Γ_5 the SE associated with node $(m + 1, n)$. The interior volume of the conservation element is also associated with node $(m + \frac{1}{2}, n + \frac{1}{2})$. Exactly the same idea can be applied to the multidimensional cases.

As can be seen, *with this staggered space-time cell-vertex solution updating strategy, no Riemann-solver-typed flux functions are needed for the interface flux.* There is no "left state" and "right state" when evaluating inter-cell fluxes as Riemann solvers do. We can see the Riemann-solver free DG-CVS method perfectly captures all flow features (shock waves, contact discontinuities, etc.) without pathological phenomena such as the expansion shock.

DG-CVS also solves moving mesh problems (Fig. 4 right) without special care. The Geometrical Conservation Law (GCL) which is vital on moving meshes is automatically satisfied by the space-time DG-CVS. This is because the mesh speed is automatically accounted for in the outward unit normal $\mathbf{n} = (n_x, n_y, n_t)$ of the space-time faces of the CE. Note that n_t is in general nonzero for moving meshes. Therefore, DG-CVS is also suitable for problems with moving boundaries and/or r-typed mesh adaptation.

3.3 Solving the Local Nonlinear Equation System

Eq. (7) is a nonlinear equation system for each space-time node. The Newton-Raphson iterative method is employed to solve the system. To apply the Newton-Raphson method, Eq. (7) is first written in the form

$$\mathbf{G}(\mathbf{s}) = \mathbf{0} \tag{8}$$

by moving the right hand side of (7) to the left hand side. According to the Newton-Raphson method, the solution can be updated iteratively by means of

$$\mathbf{s}^{(\tau+1)} = \mathbf{s}^{(\tau)} - \mathbf{J}^{-1}\mathbf{G}(\mathbf{s}^{(\tau)}) \tag{9}$$

where the superscript τ represents the iteration step. The Jacobian matrix $\mathbf{J} = \frac{\partial \mathbf{G}}{\partial \mathbf{s}}$.

To update $\mathbf{s}$ using Eq. (9), one can first compute the solution increment $\delta\mathbf{s}$ by solving

$$\mathbf{J}\delta\mathbf{s} = -\mathbf{G}(\mathbf{s}^{(\tau)}) \tag{10}$$

which is a linear equation system and can be solved by the LU-decomposition method. The solution is then updated via $\mathbf{s}^{(\tau+1)} = \mathbf{s}^{(\tau)} + \delta\mathbf{s}$.

Inside each Newton iteration, a linear equation system must be solved. To improve the efficiency, the Jacobian can be fixed during the iteration process. In the current implementation, the Jacobian is computed only once using the initial solution $\mathbf{u}^{(0)}$ before the iteration process. The Jacobian is then LU-decomposed and remain unchanged during the iteration. Inside each iteration, only the right hand side of (10) is updated and only the back substitution operations are performed. In practice, this fixed Jacobian Newton method works very well and only slightly increases the number of iterations. However, the overall efficiency is greatly improved.

3.4 Quadrature-free Implementation

As seen in Eq. (7), the DG-CVS formulation involves both surface integrals and volume integrals. An appropriate quadrature rule (e.g., Gaussian quadrature rule) is typically used to numerically integrate the surface and volume integrals. However, quadrature rules are expensive. A sufficiently large number of quadratures points must be used to ensure the accuracy. When the employed basis polynomial is of degree p, the quadrature rule must be exact for polynomials of degree $2p+1$ for surface integrals and must be exact for polynomials of degree $2p$ for volume integrals [3]. Therefore, the number of quadrature points increases rapidly with the degree of polynomials in higher dimensions. Quadrature free implementation is desirable to improve the efficiency of high order methods.

To allow the quadrature-free implementation, the spatial flux vectors must be expanded in terms of the basis polynomials similar to those used to expand the conservative variables. For example, inviscid fluxes can be expressed as

$$\mathbf{f}^h = \sum_{j=1}^{\tilde{N}} \mathbf{s}_j^{\mathbf{f}} \phi_j, \text{ and } \mathbf{g}^h = \sum_{j=1}^{\tilde{N}} \mathbf{s}_j^{\mathbf{g}} \phi_j. \tag{11}$$

where $\tilde{N}$ is the number of basis polynomials of one degree higher than those to expand the conservative variables as in Eq. (2). The requirement of using basis polynomials of degree $p+1$ is necessary to ensure the accuracy of the scheme in the case of nonlinear fluxes. The method based on the Cauchy-Kovalewski (CK) procedure [4, 5] is especially suitable for our purpose. In the current implementation of DG-CVS, the conservative variables are expanded using the Taylor polynomials. With the Taylor polynomials, the spatial and temporal derivatives of the solution are explicitly available. Using the CK procedure, the space-time derivatives of the flux vectors can be obtained using the space-time derivatives of the conservative variables.

Taylor polynomials of degree p have the following general form

$$\phi_j = c\tilde{x}^l \tilde{y}^m \tilde{t}^n \tag{12}$$

where c is a constant, $\tilde{x} = (x - x_0)/\delta x$, $\tilde{y} = (y - y_0)/\delta y$, and $\tilde{t} = (t - t_0)/\delta t$. Here (x_0, y_0, t_0) is the location of the space-time node whose solutions are to be solved. δx, δy and δt are the characteristic sizes of the local

space-time conservation element. The scaling using δx, δy and δt helps to improve the condition number of the local system. In (12), l, m, and n are integer exponents ranging from 0 to p and $0 \leq l+m+n \leq p$.

Note that, for the Taylor polynomials, the derivatives of the polynomial are a subset of the original Taylor polynomials. Therefore, the products $\phi_i\phi_j$, $\frac{\partial\phi_i}{\partial t}\phi_j$, $\frac{\partial\phi_i}{\partial x}\phi_j$ and $\frac{\partial\phi_i}{\partial y}\phi_j$ all have the forms similar to (12). This fact allows efficient implementation when evaluating integrals involving various products of the Taylor polynomials.

Such quadrature-free integration has been implemented in our DG-CVS solver to avoid the volume integral and the integral over the top surface of the CE on the left hand side of Eq. (7). The main idea is to use the divergence theorem [6] to reduce the volume integral to surface integrals and further to line integrals. And finally the analytical formulae from [7] is used to evaluate the line integrals. The efficiency of the integration is significantly increased with such quadrature-free technique. The benefit is higher for higher p.

3.4.1 Cauchy-Kovalewski Procedure

To implement quadrature-free integration, one must express the nonlinear flux vectors in terms of expansions of basis polynomials used by the state vector. The method based on the Cauchy-Kovalewski (CK) procedure[5, 4] is especially suitable for our purpose. In the current implementation of DG-CVS, the conservative variables are expanded using the Taylor polynomials. With the Taylor polynomials, the spatial and temporal derivatives of the solution are explicitly available. Using the CK procedure, the space-time derivatives of the flux vectors can be obtained using the space-time derivatives of the conservative variables.

To accomplish this goal, several methods can be used. The first possible method is based on the L_2 projection[8]. For example, to obtain the expansion of ρu^2, one may express $\rho u^2 = \frac{(\rho u)(\rho u)}{\rho}$ which leads to $\rho(\rho u^2) = (\rho u)(\rho u)$. The L_2 projection method states that

$$\int_{\Omega_K} \phi_i \rho \sum_{j=1}^{\tilde{N}} \left(s_j^{\rho u^2} \phi_j \right) \mathrm{d}\Omega = \int_{\Omega_K} \phi_i (\rho u)(\rho u) \mathrm{d}\Omega, \quad i = 1, 2, \cdots, \tilde{N} \tag{13}$$

which is an $\tilde{N} \times \tilde{N}$ linear system and can be solved for $\left\{ s_j^{\rho u^2} \right\}_{j=1}^{\tilde{N}}$. However, the left hand side matrix is not a stationary mass matrix since ρ appears in the left hand side of Eq. (13). During the iterative process, the matrix must be recomputed when ρ is updated. Therefore, the projection method is not quite efficient.

Another method is to reconstruct the flux vectors using the flux values at a nodal set [9] which is large enough to support the reconstruction of polynomials of degree $p+1$. The flux values at each is computed using the conservative variables at that node. This method is especially suitable for elements where the Lagrangian basis functions are clearly defined. With the Lagrangian polynomials, the functions can be expanded using the nodal values.

Yet another method based on the Cauchy-Kovalewski (CK) procedure[5, 4] is especially suitable for our purpose. In DG-CVS, the conservative variables are expanded using the Taylor polynomials. With the Taylor polynomials, the spatial and temporal derivatives of the solution are explicitly available. Using the CK procedure, the space-time derivatives of the flux vectors can be obtained using the space-time derivatives of the conservative variables.

Following Dyson [5] and Dumbser et al. [4], the Cauchy-Kovalewski procedure which is based on the multidimensional extension of the Leibniz rule can be formulated as

$$\mathcal{L}^{(a,b,c)}(f_1, f_2) = \frac{\partial^{a+b+c} f_1(x,y,t) f_2(x,y,t)}{\partial x^a y^b t^c} = \tag{14}$$

$$\sum_{i=0}^{a}\sum_{j=0}^{b}\sum_{k=0}^{c} \left[\begin{pmatrix} a \\ i \end{pmatrix} \begin{pmatrix} b \\ j \end{pmatrix} \begin{pmatrix} c \\ k \end{pmatrix} \frac{\partial^{(a-i)+(b-j)+(c-k)} f_1}{\partial x^{a-i} \partial y^{b-j} \partial t^{c-k}} \cdot \frac{\partial^{i+j+k} f_2}{\partial x^i \partial y^j \partial t^k} \right]$$

where $\begin{pmatrix} a \\ i \end{pmatrix}$ etc. represents the binomial coefficients.

Note that Eq. (14) is used to evaluate the space-time derivatives of the product of any two space-time functions f_1 and f_2. For example, when $f_1 = \rho u$ and $f_2 = u$, the space-time derivatives of ρu^2 can

be obtained using Eq. (14). However, the space-time derivatives of u is not available since u is not a conservative variable. Therefore, using Eq. (14) alone is not sufficient for our goal. Fortunately, Dumbser et al. [4] present the following modified Leibniz rule to obtain the space-time derivatives of f_2 from the known space-time derivatives of f_1 and $f_1 f_2$:

$$\mathcal{L}_{**}^{(a,b,c)}(f_1 f_2, f_1, f_2) = \frac{\partial^{a+b+c} f_2(x,y,t)}{\partial x^a y^b t^c} = \frac{1}{f_1}\left[\frac{\partial^{a+b+c} f_1(x,y,t) f_2(x,y,t)}{\partial x^a y^b t^c} - \mathcal{L}_{*}^{(a,b,c)}(f_1, f_2)\right] \tag{15}$$

where $\mathcal{L}_{*}^{(a,b,c)}(f_1, f_2)$ is evaluated using Eq. (14) except that the last term of the sum (i.e. $i = a$, $j = b$ and $k = c$) is set to be zero, namely,

$$\mathcal{L}_{*}^{(a,b,c)}(f_1, f_2) = \frac{\partial^{a+b+c} f_1(x,y,t) f_2(x,y,t)}{\partial x^a y^b t^c} = \sum_{i=0}^{a}\sum_{j=0}^{b}\sum_{k=0}^{c}\left[\binom{a}{i}\binom{b}{j}\binom{c}{k}\frac{\partial^{(a-i)+(b-j)+(c-k)} f_1}{\partial x^{a-i}\partial y^{b-j}\partial t^{c-k}}\cdot\frac{\partial^{i+j+k} f_2}{\partial x^i \partial y^j \partial t^k}\right] \quad \text{with } \frac{\partial^{a+b+c} f_2}{\partial x^a y^b t^c} = 0 \tag{16}$$

Therefore, to obtain the space-time derivatives of u, we just need to let $f_1 = \rho$, $f_2 = u$ and $f_1 f_2 = \rho u$ and utilize Eqs. (15) and (16).

In this paper, Eqs. (14-16) are used to construct the flux expansions. Based on the above formulations, the Cauchy-Kovalewski procedure can be summarized for the flux vectors of the Euler equations:

- Step 1: use Eqs. (15) and (16) to obtain the space-time derivatives of u and v from the known space-time derivatives of ρ, ρu and ρv.
- Step 2: use Eq. (14) to obtain the space-time derivatives of ρu^2, ρuv, and ρv^2 from the known derivatives of ρu, ρv, u and v.
- Step 3: the space-time derivatives of the pressure P can be obtained readily since

  $$P = (\gamma - 1)\left[\rho E - 0.5(\rho u^2 + \rho v^2)\right]$$

  where ρE is a conservative variable.
- Step 4: the space-time derivatives of ρH can be readily obtained via $\rho H = \rho E + P$.
- Step 5: use Eq. (14) to obtain the space-time derivatives of $\rho H u$ and $\rho H v$ from the known derivatives of ρH, u and v.

After all these five steps, all quantities in the flux vectors can be expanded in terms of the Taylor expansions.

3.4.2 Converting volume integrals to surface integrals

The integrand in the volume integral (cf. (7)) has the following general form

$$q(x,y,t) = q_1(x,y)\Delta x^l \Delta y^m \Delta t^n \tag{17}$$

which results from the products between Taylor polynomials. Here, $q_1(x,y)$ is a non-polynomial spatial function. For instance, in some applications, the source term cannot be explicitly expressed as an polynomial function. So $q_1(x,y)$ is kept here without loss of generality. In (17), l, m, and n are integer exponents. For notational simplicity, denote $\tilde{x} = \Delta x = x - x_0$, $\tilde{y} = \Delta y = y - y_0$, and $\tilde{t} = \Delta t = t - t_0$. Then (17) becomes

$$q(x,y,t) = q_1(x,y)\tilde{x}^l \tilde{y}^m \tilde{t}^n \tag{18}$$

Computing volume integrals with integrands of the form as in (18) is equivalent to computing the moments of arbitrary order of polyhedra. In the current case, the polyhedra are polygonal cylinders. The method of using the divergence theorem explained in [6] is adopted to reduce the volume integral to surface integrals.

To evaluate a scalar integral inside a space-time volume Ω, i.e. $\int_\Omega q\mathrm{d}\Omega$, we first construct a vector function $\mathbf{Q}(x,y,t) = Q_1\mathbf{i}_1 + Q_2\mathbf{i}_2 + Q_3\mathbf{i}_3$ where $\mathbf{i}_1$, $\mathbf{i}_2$ and $\mathbf{i}_3$ denote the unit vectors along the x, y, and t-directions, respectively, such that

$$q = \nabla \cdot \mathbf{Q} \tag{19}$$

With the aid of the auxiliary function $\mathbf{Q}$, one can convert the volume integral into a surface integral via the divergence theorem as follows.

$$\int_\Omega q(x,y,t)\mathrm{d}\Omega = \int_\Omega \nabla\cdot\mathbf{Q}\mathrm{d}\Omega = \int_{\partial\Omega} \mathbf{Q}\cdot\mathbf{n}\mathrm{d}\Gamma \tag{20}$$

Without loss of generality, we can define the following for $\mathbf{Q}$

$$Q_1 = 0,\ Q_2 = 0,\ \text{ and } Q_3(x,y,t) = \int q(x,y,t)\mathrm{d}t + c(x,y) \tag{21}$$

to satisfy (19). In (21), $\int q\mathrm{d}t$ is an indefinite integral with respect to t and $c(x,y)$ is an arbitrary function independent of t.

Substituting (21) into (20) to obtain

$$\int_\Omega q(x,y,t)\mathrm{d}\Omega = \int_{\partial\Omega} Q_3 n_t \mathrm{d}A = \int_{\partial\Omega}\left(\int q(x,y,t)\mathrm{d}t\right) n_t\mathrm{d}A + \int_{\partial\Omega} c(x,y)n_t\mathrm{d}A \tag{22}$$

where n_t is the t-component of the outward unit normal of the surface.

$\int_{\partial\Omega} c(x,y)n_t\mathrm{d}\Gamma$ equals zero for enclosed surfaces since $c(x,y)$ is independent of t and results in zero net contributions. Therefore, it suffices to evaluate $\int_\Omega q\mathrm{d}\Omega$ via

$$\int_\Omega q(x,y,t)\mathrm{d}\Omega = \int_{\partial\Omega}\left(\int q(x,y,t)\mathrm{d}t\right) n_t\mathrm{d}A \tag{23}$$

Since $n_t = 1$ and $n_t = -1$ on the top and bottom faces of the CE, respectively, (23) can be computed as

$$\begin{aligned}\int_\Omega q(x,y,t)\mathrm{d}\Omega &= \int_{\Gamma_{\text{top}}}\left(\int q(x,y,t)\mathrm{d}t\right)\mathrm{d}A \\ &+ \int_{\Gamma_{\text{side}}}\left(\int q(x,y,t)\mathrm{d}t\right) n_t\mathrm{d}A - \int_{\Gamma_{\text{bott}}}\left(\int q(x,y,t)\mathrm{d}t\right)\mathrm{d}A\end{aligned} \tag{24}$$

Furthermore, on the top surface, $\tilde{t} = \Delta t = 0$, therefore, the first integral on the right hand side of Eq. (24) is zero. (24) reduces to

$$\int_\Omega q(x,y,t)\mathrm{d}\Omega = \int_{\Gamma_{\text{side}}}\left(\int q(x,y,t)\mathrm{d}t\right) n_t\mathrm{d}A - \int_{\Gamma_{\text{bott}}}\left(\int q(x,y,t)\mathrm{d}t\right)\mathrm{d}A \tag{25}$$

For simple polynomials, the indefinite integral can be evaluated analytically, i.e.

$$\int q(x,y,t)\mathrm{d}t = q_1(x,y)\tilde{x}^l\tilde{y}^m\int\tilde{t}^n\mathrm{d}t \quad = \quad \frac{1}{n+1}q_1(x,y)\tilde{x}^l\tilde{y}^m\tilde{t}^{n+1} \tag{26}$$

Substituting (26) into (25) and considering the fact that $\tilde{t}$ is constant on the horizontal bottom face of the CE leads to

$$\begin{aligned}\int_\Omega q(x,y,t)\mathrm{d}\Omega &= \frac{1}{n+1}\int_{\Gamma_{\text{side}}}\left(q_1(x,y)\tilde{x}^l\tilde{y}^m\tilde{t}^{n+1}\right) n_t\mathrm{d}A \\ &- \frac{1}{n+1}\tilde{t}^{n+1}\int_{\Gamma_{\text{bott}}} q_1(x,y)\tilde{x}^l\tilde{y}^m\mathrm{d}A\end{aligned} \tag{27}$$

As can be seen, the volume integral $\int_\Omega q(x,y,t)\mathrm{d}\Omega$ has been reduced to a surface integral (27) which can be integrated numerically using the Gaussian rule.

The integrals in (27) do not need to be computed in an exclusive subroutine. Instead, they can be computed in the same subroutine where the surface integrals on the right hand side of Eq. (7) are evaluated using the Gaussian quadrature rule. Therefore, the extra cost of computing (27) is trivial.

4 Major Progress #1: Solving Advection-Diffusion Equations

4.1 Grid Convergence Study

An important measurement of the performance of any high-order method is its convergence order of accuracy. In the context of finite element-based methods, when basis polynomials of degree p is used, and if the solution u and the solution gradient u_x are approximated using Eqs. (2) and (3), respectively, then the optimal convergence orders are $p+1$ for the solution u and p for the solution gradient. Theoretical analysis of convergence orders of high-order methods is difficult, it not impossible. However, the convergence order can be easily determined numerically via the grid convergence study with appropriate error norms. In this sub-section, the grid convergence study is conducted on the 1-D advection-diffusion equation and the 2-D heat equation. The convergence behavior of DG-CVS for the advection equation and the diffusion equation will be compared.

To determine the numerical convergence order, the following error norms are defined:

$$\begin{aligned} l_\infty(\epsilon) &= \max_{i=1}^{n_\mathrm{v}} |u^h(\mathbf{x}_i) - u^{\mathrm{exact}}(\mathbf{x}_i)| \\ l_2(\epsilon) &= \sqrt{\frac{1}{n_\mathrm{v}} \sum_{i=1}^{n_\mathrm{v}} \left(u^h(\mathbf{x}_i) - u^{\mathrm{exact}}(\mathbf{x}_i)\right)^2} \\ L_2(\epsilon) &= \sqrt{\frac{1}{|\Omega|} \sum_{i=1}^{n_\mathrm{v}} \int_{\Omega_i} \left(u_i^h(\mathbf{x}) - u_i^{\mathrm{exact}}(\mathbf{x})\right)^2 \mathrm{d}\Omega} \end{aligned} \tag{28}$$

where n_v is the number of vertices in the computational domain, $u^h(\mathbf{x}_i)$ is the computed numerical solution at ith vertex and $u^{\mathrm{exact}}(\mathbf{x}_i)$ is the analytical solution at the ith vertex, Ω_i is the spatial domain associated with the ith vertex and $|\Omega|$ is the size of the entire computational domain. Here, l_∞ and l_2 norms are evaluated at the discrete location of vertices and L_2 norm is obtained by integrating the continuous solution within the spatial domain associated with each vertex.

4.1.1 Convergence Orders on 1-D Meshes

The first test is to solve the following 1-D linear scalar advection-diffusion equation

$$\begin{aligned} \frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} &= 0, \qquad -1 \le x \le 1 \\ u(x,0) = u_0(x) &= \sin(\pi x), \qquad \text{periodic b.c.} \end{aligned} \tag{29}$$

The analytical solution is given as

$$u_{\mathrm{exact}} = e^{-\pi^2 \nu t} \sin(\pi(x - at))$$

The following two cases are tested:

- pure advection equation. $a = 1.0$, $\nu = 0$.
- heat equation. $a = 0$, $\nu = 1.0$.

For each case, four meshes (number of cells nc = 10, 20, 40, and 80) are used for varying degrees of basis polynomials ($p = 1$, 2, 3, and 4).

For the advection case, the time step is chosen as $\delta t = \sigma \delta x / a$ where δx is the cell interval and the Courant number $\sigma = 0.5$, 0.3125, 0.25 and 0.25 for $p1$, $p2$, $p3$ and $p4$ cases, respectively. For the stability limits in the case of advection equations, one can refer to our earlier paper[10]. All cases are computed up to $t = 1.0$. The numerical convergence orders using the l_∞, l_2 and L_2 norms are recorded in Tables 2, 3 and 4. All three tables show that the numerical convergence orders are $p+1$ for u and p for u_x for all p's. The convergence orders are optimal.

For the diffusion case, the time step is chosen as $\delta t = 0.1\delta x^2/\nu$ for all $p1 - p4$ cases. All cases are computed up to $t = 0.1$. The numerical convergence orders based on various norms are recorded in Tables 5

Table 2: 1-D advection equation. $a = 1.0$, $t = 1.0$. Numerical convergence order determined by l_∞ norm.

		nc = 10	nc = 20		nc = 40		nc = 80		
p	variable	$l_\infty(\epsilon)$	$l_\infty(\epsilon)$	order	$l_\infty(\epsilon)$	order	$l_\infty(\epsilon)$	order	comments
1	u	2.11E-02	4.88E-03	2.113	1.13E-03	2.112	2.70E-04	2.064	optimal
	u_x	1.08E-01	4.38E-02	1.301	2.09E-02	1.071	1.03E-02	1.016	optimal
2	u	2.13E-04	2.77E-05	2.946	3.51E-06	2.979	4.41E-07	2.995	optimal
	u_x	2.89E-02	7.18E-03	2.008	1.79E-03	2.001	4.48E-04	2.000	optimal
3	u	2.56E-05	1.58E-06	4.019	9.71E-08	4.023	6.04E-09	4.007	optimal
	u_x	4.04E-04	6.03E-05	2.742	7.80E-06	2.951	9.83E-07	2.989	optimal
4	u	4.18E-07	1.28E-08	5.031	3.90E-10	5.037	1.25E-11	4.965	optimal
	u_x	5.59E-05	3.51E-06	3.992	2.21E-07	3.991	1.39E-08	3.990	optimal

Table 3: 1-D advection equation. $a = 1.0$, $t = 1.0$. Numerical convergence order determined by l_2 norm.

		nc = 10	nc = 20		nc = 40		nc = 80		
p	variable	$l_2(\epsilon)$	$l_2(\epsilon)$	order	$l_2(\epsilon)$	order	$l_2(\epsilon)$	order	comments
1	u	1.48E-02	3.36E-03	2.134	7.88E-04	2.095	1.90E-04	2.055	optimal
	u_x	7.71E-02	3.07E-02	1.330	1.46E-02	1.071	7.25E-03	1.010	optimal
2	u	1.61E-04	2.02E-05	2.994	2.52E-06	3.006	3.14E-07	3.006	optimal
	u_x	2.13E-02	5.20E-03	2.037	1.28E-03	2.018	3.19E-04	2.009	optimal
3	u	1.76E-05	1.09E-06	4.013	6.79E-08	4.009	4.24E-09	3.999	optimal
	u_x	2.86E-04	4.18E-05	2.774	5.45E-06	2.940	6.91E-07	2.981	optimal
4	u	3.12E-07	9.28E-09	5.069	2.79E-10	5.057	8.93E-12	4.964	optimal
	u_x	4.13E-05	2.54E-06	4.022	1.58E-07	4.008	9.89E-09	3.999	optimal

Table 4: 1-D advection equation. $a = 1.0$, $t = 1.0$. Numerical convergence order determined by L_2 norm.

		nc = 10	nc = 20		nc = 40		nc = 80		
p	variable	$L_2(\epsilon)$	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	comments
1	u	1.17E-02	2.79E-03	2.064	6.87E-04	2.020	1.71E-04	2.006	optimal
	u_x	4.04E-01	2.03E-01	0.993	1.02E-01	0.998	5.09E-02	0.999	optimal
2	u	5.73E-04	7.21E-05	2.990	9.03E-06	2.997	1.13E-06	2.999	optimal
	u_x	3.66E-02	9.21E-03	1.990	2.31E-03	1.998	5.77E-04	1.999	optimal
3	u	2.55E-05	1.64E-06	3.961	1.03E-07	3.988	6.47E-09	3.997	optimal
	u_x	2.31E-03	2.94E-04	2.975	3.69E-05	2.992	4.62E-06	2.998	optimal
4	u	8.02E-07	2.50E-08	5.005	7.80E-10	5.002	2.46E-11	4.986	optimal
	u_x	1.00E-04	6.25E-06	3.999	3.90E-07	4.002	2.45E-08	3.996	optimal

Table 5: 1-D heat equation. $\nu = 1.0$, $t = 0.1$. Numerical convergence order determined by l_∞ norm.

		nc = 10	nc = 20		nc = 40		nc = 80		
p	variable	$l_\infty(\epsilon)$	$l_\infty(\epsilon)$	order	$l_\infty(\epsilon)$	order	$l_\infty(\epsilon)$	order	comments
1	u	9.72E-03	2.62E-03	1.894	6.58E-04	1.991	1.65E-04	1.998	optimal
	u_x	3.04E-02	7.82E-03	1.959	1.97E-03	1.990	4.93E-04	1.997	super-optimal
2	u	1.32E-03	3.70E-04	1.838	9.40E-05	1.977	2.36E-05	1.994	sub-optimal
	u_x	3.17E-02	8.28E-03	1.936	2.09E-03	1.983	5.25E-04	1.996	optimal
3	u	7.29E-05	5.10E-06	3.836	3.24E-07	3.976	2.03E-08	3.994	optimal
	u_x	5.08E-05	3.66E-06	3.792	2.37E-07	3.948	1.50E-08	3.987	super-optimal
4	u	5.14E-06	3.23E-07	3.993	1.99E-08	4.020	1.24E-09	4.006	sub-optimal
	u_x	2.79E-05	1.77E-06	3.974	1.11E-07	3.992	6.98E-09	3.998	optimal

Table 6: 1-D heat equation. $\nu = 1.0$, $t = 0.1$. Numerical convergence order determined by l_2 norm.

		nc = 10	nc = 20		nc = 40		nc = 80		
p	variable	$l_2(\epsilon)$	$l_2(\epsilon)$	order	$l_2(\epsilon)$	order	$l_2(\epsilon)$	order	comments
1	u	6.89E-03	1.81E-03	1.933	4.60E-04	1.974	1.16E-04	1.989	optimal
	u_x	2.25E-02	5.66E-03	1.988	1.41E-03	2.006	3.51E-04	2.006	super-optimal
2	u	9.38E-04	2.55E-04	1.877	6.57E-05	1.959	1.66E-05	1.985	sub-optimal
	u_x	2.34E-02	5.99E-03	1.965	1.50E-03	2.000	3.74E-04	2.004	optimal
3	u	5.16E-05	3.52E-06	3.875	2.26E-07	3.959	1.43E-08	3.985	optimal
	u_x	3.75E-05	2.65E-06	3.822	1.70E-07	3.964	1.07E-08	3.995	super-optimal
4	u	3.65E-06	2.23E-07	4.032	1.39E-08	4.002	8.71E-10	3.997	sub-optimal
	u_x	2.06E-05	1.28E-06	4.003	7.98E-08	4.008	4.97E-09	4.006	optimal

and 6 and 7. Comparing with the optimal convergence rates for the advection equation, one can see that the convergence rates for the diffusion equation appear inconsistent. For clarity's sake, the following observations from Tables 5 and 6 and 7 are summarized:

- When p is odd,
  - the convergence rates based on the l_∞ and l_2 norms are optimal for u and super-optimal for u_x.
  - the convergence rates based on the L_2 norm are optimal for both u and u_x.
- When p is even, the convergence rates based on all norms are sub-optimal for u and optimal for u_x.

Since l-norms do not produce the same convergence orders for the heat equation as those for the advection equation no matter whether p is odd or even, while the convergence orders based on the L_2-norm are optimal for both the heat equation and the advection equation when p is odd, this indicates that L_2-norm is a more appropriate norm for determining the convergence order. We will focus on L_2-norm in the remaining tests.

The inconsistent convergence behavior between odd degree and even degree approximations has also been reported in the methods of many other researcher [11, 12, 13, 14, 15]. It is also interesting to note that the first version of the central LDG method by Liu et al.[16] is sub-optimal first order accurate for $p = 1$ and optimal $(p + 1)$th order accurate for $p > 1$.

4.1.2 Convergence Orders on 2-D Structured and Unstructured Meshes

To investigate the convergence behavior of DG-CVS for 2-D diffusion equations, rectangular and unstructured triangular meshes with various resolutions are used. Figure 5 shows the coarsest rectangular and unstructured triangular meshes used in the test. The coarsest rectangular mesh is composed of 10×10 rectangular cells and is designated as "qua-10". The coarsest triangular mesh is designated as "tri-10" whose edge resolution is comparable to that of mesh qua-10. The meshes will be refined isotropically several times in the grid convergence study, resulting in a series of meshes designated as "qua-20", "qua-40", "qua-80", "tri-20", "tri-40" and "tri-80", respectively.

Table 7: 1-D heat equation. $\nu = 1.0$, $t = 0.1$. Numerical convergence order determined by L_2 norm.

		nc = 10	nc = 20		nc = 40		nc = 80		
p	variable	$L_2(\epsilon)$	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	comments
1	u	4.90E-03	1.24E-03	1.982	3.11E-04	1.995	7.78E-05	1.999	optimal
	u_x	1.49E-01	7.50E-02	0.994	3.75E-02	0.999	1.88E-02	1.000	optimal
2	u	1.22E-03	2.78E-04	2.130	6.75E-05	2.042	1.68E-05	2.011	sub-optimal
	u_x	1.54E-02	3.93E-03	1.965	9.90E-04	1.991	2.48E-04	1.998	optimal
3	u	3.70E-05	2.45E-06	3.918	1.55E-07	3.978	9.76E-09	3.994	optimal
	u_x	6.50E-04	8.09E-05	3.006	1.01E-05	3.000	1.26E-06	3.000	optimal
4	u	3.92E-06	2.30E-07	4.094	1.41E-08	4.025	8.77E-10	4.007	sub-optimal
	u_x	3.75E-05	2.15E-06	4.128	1.31E-07	4.034	8.14E-09	4.008	optimal

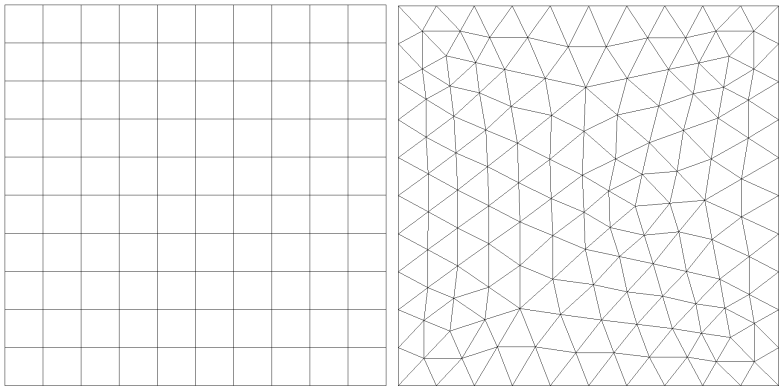


Figure 5: Two-dimensional meshes used in the tests. Left: rectangular mesh. Right: unstructured triangular mesh.

Table 8: 2-D heat equation with sinusoidal initial solution on rectangular meshes. $\nu = 1.0$, $t = 0.1$. Numerical convergence order determined by the L_2 norm.

		qua-10	qua-20		qua-40		qua-80		
p	variable	$L_2(\epsilon)$	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	comments
1	u	4.27e-03	1.09e-03	1.970	2.73e-04	1.997	6.84e-05	1.997	optimal
	u_x and u_y	7.81e-02	3.94e-02	0.987	1.98e-02	0.993	9.89e-03	1.001	optimal
2	u	8.37e-04	2.02e-04	2.051	5.00e-05	2.014	1.25e-05	2.000	sub-optimal
	u_x and u_y	1.24e-02	3.21e-03	1.950	8.11e-04	1.985	2.03e-04	1.998	optimal
3	u	5.43e-05	3.73e-06	3.864	2.39e-07	3.964	1.51e-08	3.984	optimal
	u_x and u_y	1.28e-03	1.58e-04	3.018	1.97e-05	3.004	2.47e-06	2.996	optimal
4	u	2.92e-06	1.68e-07	4.119	1.05e-08	4.000	6.54e-10	4.005	sub-optimal
	u_x and u_y	1.01e-04	6.06e-06	4.059	3.75e-07	4.014	2.34e-08	4.002	optimal

Table 9: 2-D heat equation with sinusoidal initial solution on triangular meshes. $\nu = 1.0$, $t = 0.1$. Numerical convergence order determined by the L_2 norm.

		tri-10	tri-20		tri-40		tri-80		
p	variable	$L_2(\epsilon)$	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	comments
1	u	4.82e-03	1.25e-03	1.947	3.22e-04	1.957	8.12e-05	1.988	optimal
	u_x	7.26e-02	3.64e-02	0.996	1.82e-02	1.000	9.13e-03	0.995	optimal
	u_y	7.25e-02	3.64e-02	0.994	1.82e-02	1.000	9.13e-03	0.995	optimal
2	u	1.37e-03	3.46e-04	1.985	8.65e-05	2.000	2.17e-05	1.995	sub-optimal
	u_x	9.07e-03	2.30e-03	1.979	5.80e-04	1.988	1.46e-04	1.990	optimal
	u_y	9.03e-03	2.29e-03	1.979	5.78e-04	1.986	1.45e-04	1.995	optimal
3	u	3.14e-05	2.02e-06	3.958	1.28e-07	3.980	8.07e-09	3.987	optimal
	u_x	7.30e-04	9.23e-05	2.983	1.16e-05	2.992	1.46e-06	2.990	optimal
	u_y	7.30e-04	9.23e-05	2.983	1.16e-05	2.992	1.46e-06	2.990	optimal
4	u	2.15e-06	1.33e-07	4.015	8.35e-09	3.994	5.27e-10	3.986	sub-optimal
	u_x	4.62e-05	2.96e-06	3.964	1.88e-07	3.977	1.19e-08	3.982	optimal
	u_y	4.62e-05	2.96e-06	3.964	1.88e-07	3.977	1.19e-08	3.982	optimal

The following 2-D heat equation with sinusoidal initial solution is solved using DG-CVS.

$$\begin{aligned} \frac{\partial u}{\partial t} - \nu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) &= 0, \qquad -1 \le x, y \le 1 \\ u(x, y, 0) &= \sin(\pi(x + y)), \qquad \text{periodic b.c.} \end{aligned} \tag{30}$$

The analytical solution is given as

$$u_{\text{exact}} = e^{-2\pi^2 \nu t} \sin(\pi(x + y))$$

The time step is chosen to be $\delta t = \sigma h^2/\nu$ where h is the local characteristic size of the element and $\sigma = 0.2$ for $p1$ cases and $\sigma = 0.1$ for all other cases. Table 8 and Table 9 show the convergence orders on rectangular meshes and triangular meshes, respectively. Both tables show the same convergence rates which indicates the convergence orders are independent on the type of the spatial mesh. In addition, the same convergence rates as those in Table 7 for the 1-D heat equation are observed, that is, the convergence orders are optimal for both u and its gradient when p is odd, and sub-optimal for u and optimal for u's gradients when p is even.

4.2 More Tests on 2D Scalar Advection-Diffusion Equation

In this sub-section, we provide two more test cases. Both cases involve time dependent boundary conditions. The first case is the 2-D heat equation with the delta initial solution, and the second case is the 2-D nonlinear viscous Burgers equation.

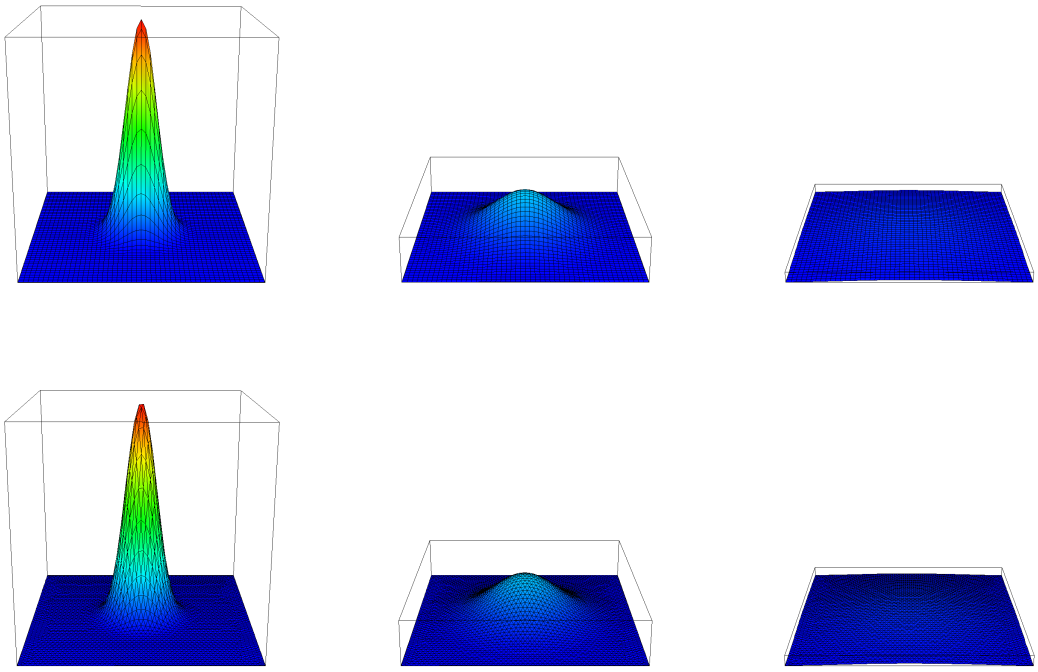


Figure 6: Solution of 2-D heat equation with the delta initial solution. Top row: solutions on the rectangular mesh qua-50 and bottom row: solutions on the triangular mesh tri-50. Left column: solution at $t = 0.01$, middle column: solution at $t = 0.05$ and right column: solution at $t = 0.2$.

From the previous tests, one can conclude that the convergence orders based on the L_2 norm is optimal for both the solution and its gradients when p is odd. Besides, this optimality holds for both the advection and diffusion equations. Since in practice, the governing equations often involve both advection and diffusion simultaneously, it justifies to employ basis polynomials of odd degrees for best and consistent convergence rates. Therefore, in the remaining test cases, only results of odd p will be presented.

4.2.1 2-D Heat Equation with the Delta Initial Solution

This case is to solve the following 2-D heat equation using DG-CVS.

$$\begin{aligned} \frac{\partial u}{\partial t} - \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) &= 0, \qquad -1 \le x, y \le 1 \\ u(x, y, 0) &= u_0 \end{aligned} \tag{31}$$

where u_0 is the delta function at the origin of the domain $(0, 0)$. The solution to Eq. (31) with such an initial condition is called the fundamental solution of the heat equation[17]. The analytical solution is given as

$$u_{\text{exact}} = \frac{1}{4\pi t} e^{\left(-\frac{x^2+y^2}{4t}\right)}$$

The boundary conditions on the four boundaries are time varying depending on the above analytical solution formula.

The time step is chosen to be $\delta t = \sigma h^2$ where h is the local characteristic size of the element and $\sigma = 0.2$ for the $p1$ case and $\sigma = 0.1$ for the $p3$ case. Figure 6 shows the carpet view of the solution at three instants, $t = 0.01$, $t = 0.05$, and $t = 0.2$, on both the rectangular and triangular meshes, respectively.

Table 10: 2-D heat equation with delta initial solution on rectangular meshes. $t = 0.05$. Numerical convergence order determined by L_2 norm.

		qua-10	qua-20		qua-40		qua-80		
p	variable	$L_2(\epsilon)$	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	comments
1	u	1.20e-02	3.12e-03	1.943	7.88e-04	1.985	1.98e-04	1.993	optimal
	u_x and u_y	2.53e-01	1.28e-01	0.983	6.43e-02	0.993	3.22e-02	0.998	optimal
3	u	4.53e-04	3.33e-05	3.766	2.19e-06	3.927	1.38e-07	3.988	optimal
	u_x and u_y	6.82e-03	7.78e-04	3.132	9.38e-05	3.052	1.15e-05	3.028	optimal

Table 11: 2-D heat equation with delta initial solution on triangular meshes. $t = 0.05$. Numerical convergence order determined by L_2 norm.

		tri-10	tri-20		tri-40		tri-80		
p	variable	$L_2(\epsilon)$	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	comments
1	u	1.15e-02	3.02e-03	1.929	7.71e-04	1.970	1.95e-04	1.983	optimal
	u_x	2.23e-01	1.12e-01	0.994	5.64e-02	0.990	2.83e-02	0.995	optimal
	u_y	2.32e-01	1.17e-01	0.988	5.85e-02	1.000	2.93e-02	0.998	optimal
3	u	1.72e-04	1.11e-05	3.954	6.99e-07	3.989	4.37e-08	4.000	optimal
	u_x	4.12e-03	5.00e-04	3.043	6.17e-05	3.019	7.69e-06	3.004	optimal
	u_y	4.46e-03	5.42e-04	3.041	6.71e-05	3.014	8.35e-06	3.006	optimal

Tables 10 and 11 show the convergence rates of $p1$ and $p3$ approximations on rectangular and triangular meshes, respectively. Not surprisingly, the convergence rates are optimal for all cases.

In Fig. 7, the solution u and its gradient u_x along the horizontal line $y = 0$ are shown together with the analytical solutions at $t = 0.05$. To visually compare the accuracy between $p1$ and $p3$ results, we intentionally choose a coarse 20×20 mesh. As can be seen, the $p1$ solution is not accurate enough to resolve the local extrema. By contrast, the $p3$ solution lies on top of the analytical solution.

4.2.2 2-D Viscous Burgers Equation

Finally, the following 2-D viscous Burgers equation is solved using DG-CVS.

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + u\frac{\partial u}{\partial y} - \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = 0, \quad 0 \le x,\, y \le 25 \tag{32}$$

with the analytical solution given as

$$u_{\text{exact}} = \frac{2}{e^{\frac{x - x_c + y - y_c - 2t}{\nu}} + 1}$$

where $(x_c, y_c) = (0, 0)$ is a constant location.

The 1-D version of this case was presented in [18]. This case is constructed such that the original wave is propagated without changing shape under the effect of both nonlinear advection and linear diffusion. The initial solution at $t = 0$ and the boundary conditions on all four boundaries are provided by the analytical solution. Therefore the boundary conditions are time dependent.

We first conduct the grid convergence study on this nonlinear advection-diffusion case. In the study, $\nu = 2.5$ is chosen and the simulation is run up to $t = 10$. In the current study, the time step is chosen as $\delta t = \sigma \min(h/a, h^2/\nu)$ where h is the local mesh size and $\sigma = 0.2$ for the $p1$ case and $\sigma = 0.1$ for the $p3$ case. For the purpose of determining the convergence rates, this choice of time steps may not be appropriate since advection and diffusion have different time scales. A more appropriate approach may be the operator splitting method where the advection and the diffusion are treated with different time steps. Tables 12 and 13 show the convergence rates of $p1$ and $p3$ on rectangular and triangular meshes, respectively. The convergence rates, though not as neat as those for the pure diffusion equation, are still close to optimal.

To further demonstrate the accuracy of DG-CVS for various values of ν, the following three cases are simulated on both qua-50 and tri-50 meshes where the size of the mesh is indicated by $\delta x = 0.5$:

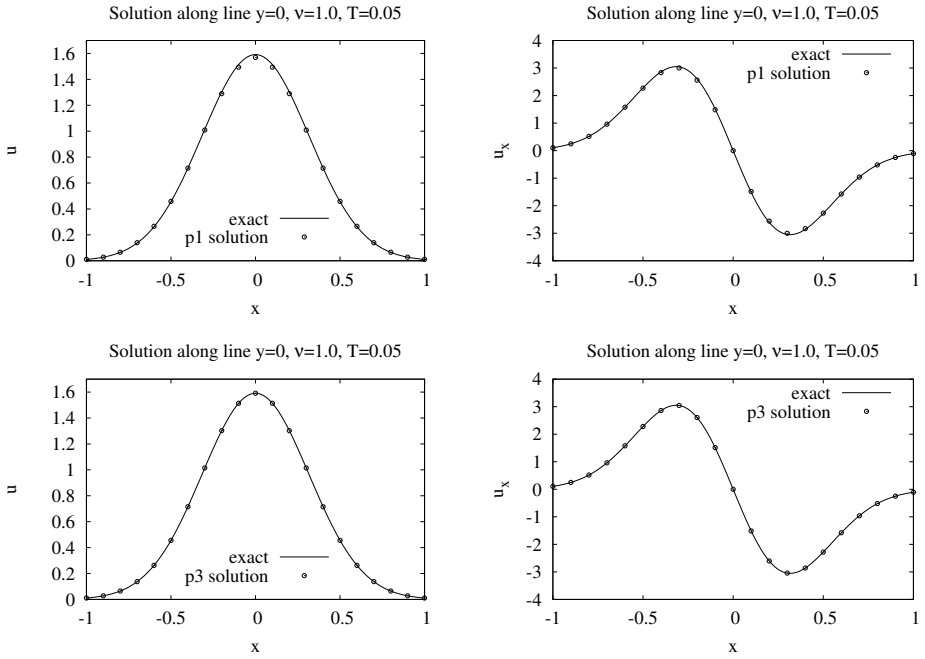


Figure 7: Comparison between $p1$ and $p3$ solutions on the 20×20 mesh at $y = 0$ and $t = 0.05$ for the case of the fundamental solution of the 2-D heat equation.

Table 12: Solution of 2-D viscous Burgers equation on rectangular meshes. $\nu = 2.5$, $t = 10$. Numerical convergence order determined by L_2 norm.

		qua-10	qua-20		qua-40		qua-80		
p	variable	$L_2(\epsilon)$	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	comments
1	u	1.38e-02	3.15e-03	2.131	7.71e-04	2.031	1.92e-04	2.006	optimal
	u_x and u_y	1.74e-02	8.52e-03	1.030	4.21e-03	1.017	2.09e-03	1.010	optimal
3	u	2.83e-04	1.69e-05	4.066	1.11e-06	3.928	7.88e-08	3.816	optimal
	u_x and u_y	7.11e-04	9.12e-05	2.963	1.19e-05	2.938	1.61e-06	2.886	optimal

Table 13: Solution of 2-D viscous Burgers equation on triangular meshes. $\nu = 2.5$, $t = 10$. Numerical convergence order determined by L_2 norm.

		tri-10	tri-20		tri-40		tri-80		
p	variable	$L_2(\epsilon)$	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	$L_2(\epsilon)$	order	comments
1	u	8.08e-03	2.19e-03	1.883	5.49e-04	1.996	1.38e-04	1.992	optimal
	u_x	1.54e-02	7.79e-03	0.983	3.86e-03	1.013	1.93e-03	1.000	optimal
	u_y	1.54e-02	7.79e-03	0.983	3.86e-03	1.013	1.93e-03	1.000	optimal
3	u	1.70e-04	1.21e-05	3.812	1.00e-06	3.597	1.03e-07	3.279	optimal
	u_x	4.16e-04	5.31e-05	2.970	6.44e-06	3.044	7.96e-07	3.016	optimal
	u_y	4.16e-04	5.34e-05	2.962	6.48e-06	3.043	8.02e-07	3.014	optimal

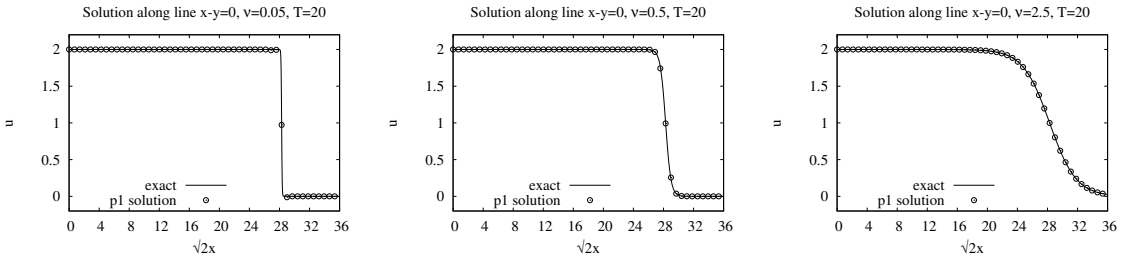


Figure 8: $p1$ solution of 2-D viscous Burgers equation along the diagonal line $x - y = 0$ at $t = 20$. Left: $\nu = 0.05$, middle: $\nu = 1.0$, and right: $\nu = 2.5$.

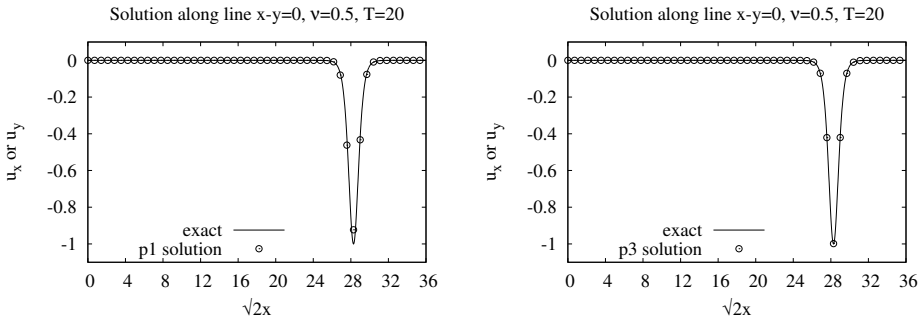


Figure 9: u_x or u_y of the 2-D viscous Burgers equation with $\nu = 0.5$ along the diagonal line at $t = 20$. Left: $p1$ solution and right: $p3$ solution.

- short wave ($\delta x/\nu = 10$, i.e. $\nu = 0.05$).
- medium wave ($\delta x/\nu = 1$, i.e. $\nu = 0.5$).
- long wave ($\delta x/\nu = 0.2$, i.e. $\nu = 2.5$).

Figure 8 plots the $p1$ solution along the diagonal line $x - y = 0$ together with the exact solution. As can be seen, all types of waves have been captured accurately in terms of both location and magnitude. In the case of the short wave, no limiter is applied and therefore slight overshoot and undershoot can be seen. The $p3$ solution is not shown since no much visual difference can be seen between the $p3$ solution and the $p1$ solution. Figure 9 compares the computed solution gradients (u_x or u_y) with the exact solution for the case of $\nu = 0.5$. As can be seen, the $p3$ solution is superior to the $p1$ solution in resolving the local sharp extremum of the solution gradient.

5 Major Progress #2: Solving Euler Equations Involving Curved Boundaries

Progress has been made to solve flow problems involving curved boundaries. Here two examples are presented to demonstrate the capability of handling curved boundaries.

5.1 Flow around a Circular Cylinder

The first example is a subsonic flow with $M = 0.1$ around a circular cylinder. The far-field boundary is located at 30 diameters from the center of the circle. The mesh contains 24×10 quadrilateral cells. Figure 10 shows the Mach number distributions from the second-order ($p1$), third order ($p2$) and fourth order ($p3$) simulations, respectively. Obviously, high-order solutions are superior to lower order ones on the same coarse mesh. Figure 11 compares the steady-state pressure distribution on the cylinder surface with the analytical solution. Again, $p3$ solution is the most accurate.

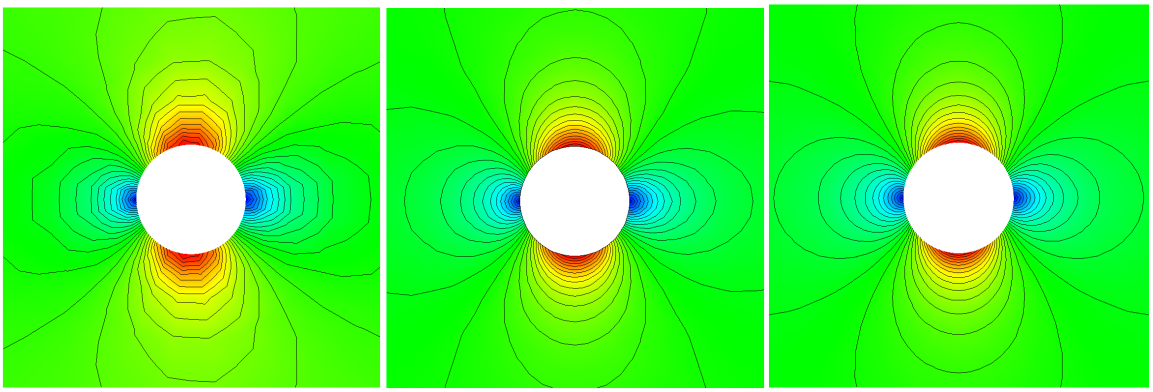


Figure 10: Mach number field of the subsonic flow ($M = 0.1$) around a circular cylinder on a 24×10 quadrilateral mesh. Left: $p1$ solution, middle: $p2$ solution, and right: $p3$ solution.

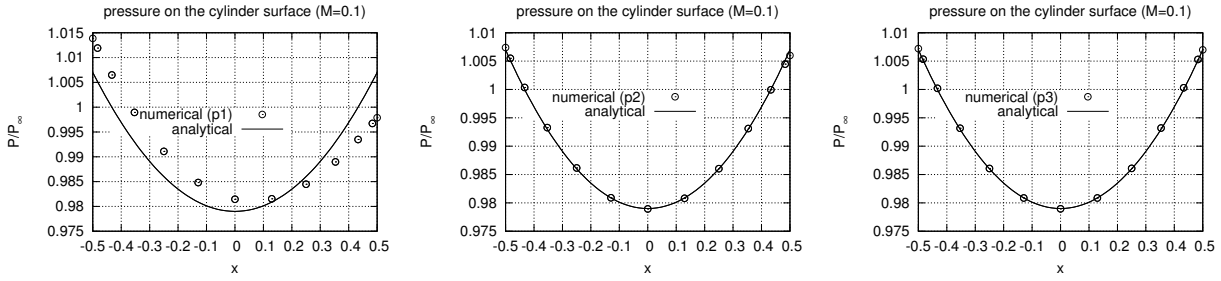


Figure 11: Pressure on the surface of the circular cylinder on a 24×10 quadrilateral mesh. Left: $p1$ solution, middle: $p2$ solution, and right: $p3$ solution.

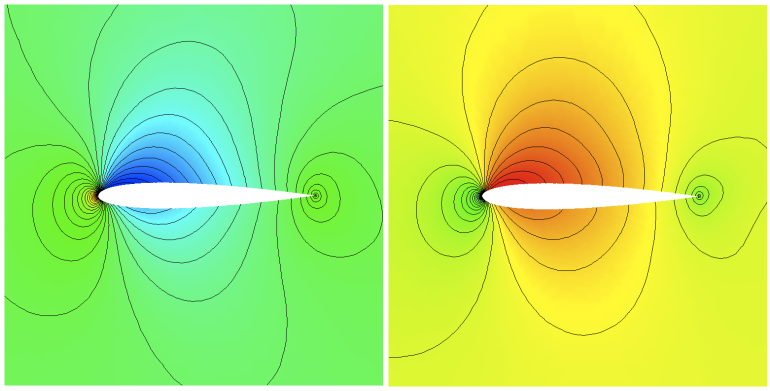


Figure 12: Subsonic flow around NACA0012 airfoil. Left: pressure field. Right: Mach number field.

5.2 Flow around NACA0012 Airfoil

The second example is a subsonic flow with Mach 0.63, AoA = 2° around the NACA0012 airfoil with curved boundary. The mesh contains 1536 quadrilateral cells with 64 cells on the airfoil surface. Figure 12 shows the $p3$ solution. As can be seen, the contour lines are smooth near the boundary including the trailing edge.

6 Major Progress #3: Solving Shallow Water Equations

The shallow water equations (SWE) are widely used as the mathematical model to numerically simulate the dam break, river inundation, failure of levees and tide of ocean in coastal and civil engineering. When the characteristic vertical velocity is small in comparison with the characteristic horizontal velocity, which happens when the characteristic vertical length scale is much smaller than the characteristic horizontal length scale, the incompressible Navier-Stokes equations can be simplified to the shallow water equations (SWE) by depth averaging the NSE in the vertical direction.

The frictionless shallow water equation can be expressed in the following conservative form

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} = \mathbf{r}(\mathbf{u}) \tag{33}$$

in which the conservative vector, flux vectors and source vector are

$$\mathbf{u} = \begin{bmatrix} H \\ Hu \\ Hv \end{bmatrix}, \mathbf{f} = \begin{bmatrix} Hu \\ Hu^2 + gH^2/2 \\ Huv \end{bmatrix}, \mathbf{g} = \begin{bmatrix} Hv \\ Huv \\ Hv^2 + gH^2/2 \end{bmatrix}, \text{ and } \mathbf{r} = \begin{bmatrix} 0 \\ gHS_{0x} \\ gHS_{0y} \end{bmatrix}$$

respectively. Here, the total water depth $H(\mathbf{x}, t) = \zeta(\mathbf{x}, t) + d(\mathbf{x})$ where $\zeta(\mathbf{x}, t)$ is the free-surface elevation and $d(\mathbf{x})$ is the still water depth. u and v are the x- and y-component of the depth-averaged velocity. g is the acceleration due to gravity. S_{0x} and S_{0y} are the bottom slopes in x- and y-directions, respectively.

Equation (33) is a hyperbolic nonlinear system. There is close mathematical and physical analogy between the shallow water flows and compressible flows. The hydraulic jumps and bores are analogous to the stationary and moving shock waves in compressible gas flows. Therefore, the numerical methods used to solve the SWE often mimic those for solving the compressible Euler equations.

The current DG-CVS method provides a Riemann-solver-free alternative approach to solve the shallow water equations. Here, several numerical examples are provided to test the accuracy of the DG-CVS based solver on solving the shallow water equations. In all the simulations, the reference velocity is chosen according to

$$U_{\text{ref}} = \sqrt{gL_{\text{ref}}}$$

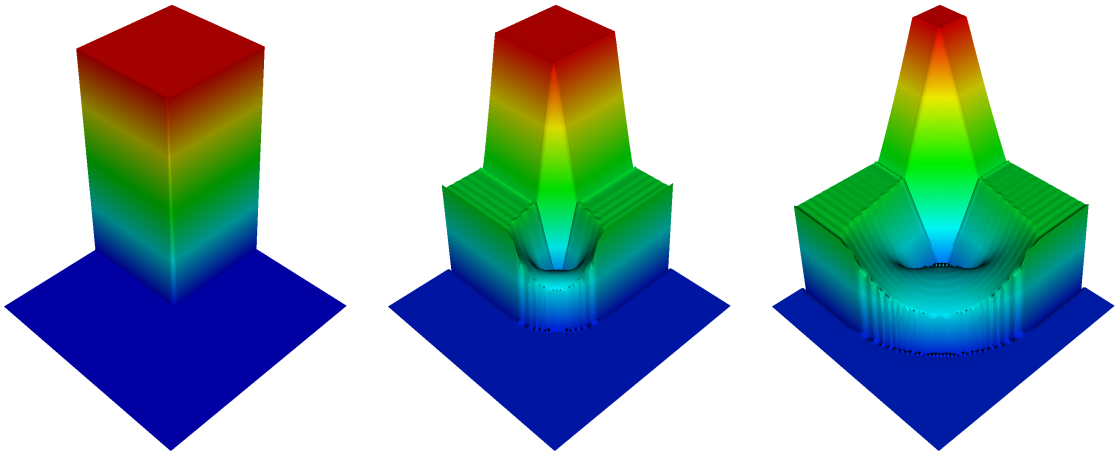


Figure 13: Water elevation carpet view of the 2-d shocktube problem. Left: $t = 0.0$, middle: $t = 1.0$ and right: $t = 2.0$.

where g is the acceleration due to gravity and L_{ref} is some reference length. Nondimensionalization based on these quantities leads to that the nondimensional g is unity. In addition, all simulations assume flat water bed.

6.1 2-D Dam Break Problem

The first case is a two-dimensional dam break problem which is analogous to the shock tube problem used to verify a compressible Euler solver. The computational domain is a $[-5, 5] \times [-5, 5]$ rectangular region. At $t = 0$, the water height is 3 within the lower left corner $[-5, 0] \times [-5, 0]$, and the rest of the domain is filled with water of height 1. The left and the bottom boundaries are symmetry ones. Figure 13 shows the carpet view of the water elevation at three time instants. The solution is obtained using the fourth order basis functions on the 99×99 mesh. As can be seen, the complex wave structures have been captured in the simulation. Note that no any type of solution limiting is employed in the simulation. Therefore, slight wiggles can be seen behind the water jump. Figure 14 shows the water elevation and momentum along one of the symmetry boundaries at times up to $t = 2.0$. The water jump discontinuity is captured sharply.

6.2 Gaussian Pulse Problem

The second case is to simulate the evolution of a Gaussian water depth perturbance. This problem is indeed one-dimensional but is simulated on a 2-D computational domain as shown in Fig. 15. The fourth order ($p3$) simulation is performed again. Figures 15 and 16 shows similar information to those in Figs. 13 and 14 in the previous example. As can be seen, the two waves propagate to two opposite directions and get steeper and steeper.

6.3 2-D Dam Break Problem

The last example is a more realistic classical benchmark problem to verify a shallow water equation solver. Figure 17 depicts the domain geometry. Initially, two reservoirs are separated by a lock gate which is of 75 meters wide. The water levels are 10 meters and 5 meters, respectively. The reservoir on the left has higher water level. At $t = 0$, the lock gate is opened. The evolution of the water level is simulated using the second order ($p1$) DG-CVS solver. The resolution of the computational mesh is 2.5 meters. Figure 18 shows the contour lines and carpet view of the water level at $t = 7.2$ seconds.

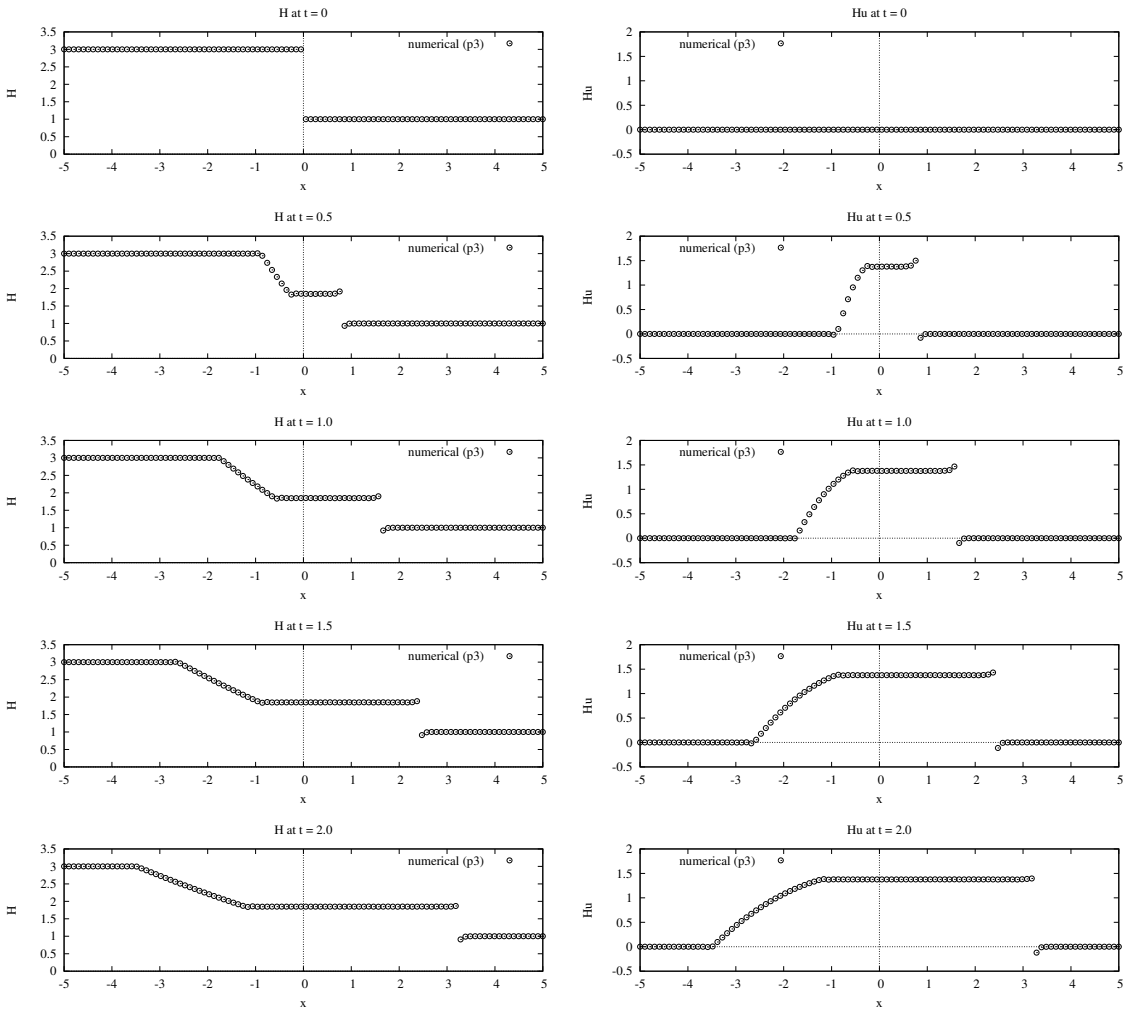


Figure 14: Water element and momentum along the symmetry line at various instants. Left: water elevation and right: momentum.

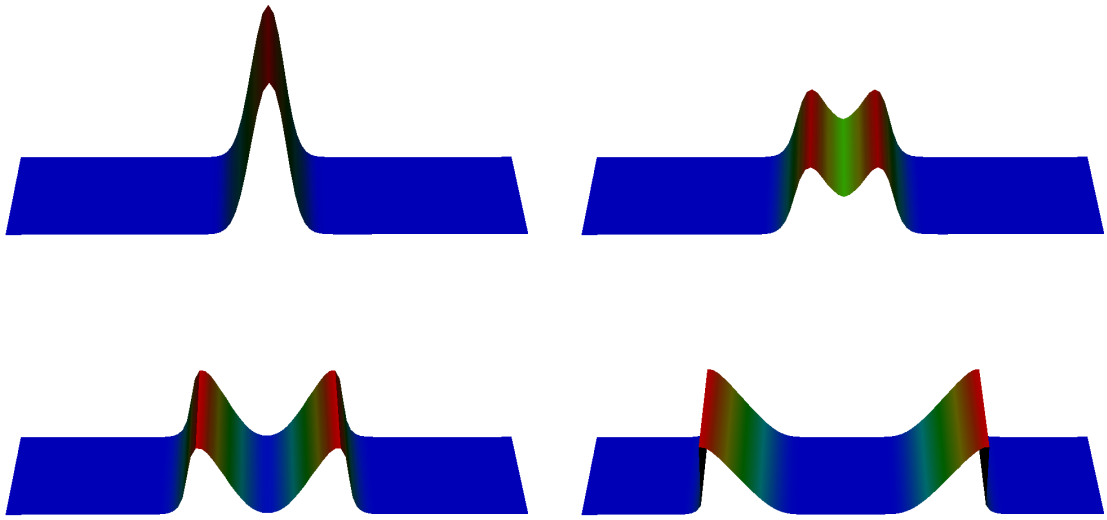


Figure 15: Carpet view of the evolution of Gaussian pulse depth perturbation. Clockwisely, $t = 0$, $t = 0.5$, $t = 1.0$ and $t = 2.0$.

7 Major Progress #4: Solving Level Set Equation

The DG-CVS solver has been also extended to solve the level-set equation to accurately resolve the interface between fluids. In the level set method, the interface is represented by the zero level set $\mathcal{C}(t) = \{\mathbf{x}|\psi(\mathbf{x}, t) = 0\}$ of a level set function $\psi(\mathbf{x}, t)$. ψ is often taken to be the signed distance function to the interface. The level set equation can be expressed as

$$\psi_t + \mathbf{v} \cdot \nabla\psi = 0 \tag{34}$$

where $\mathbf{v}$ is the velocity field which evolves the interface. Equation (34) can be rewritten in conservative form

$$\psi_t + \nabla \cdot (\psi\mathbf{v}) = \psi\nabla \cdot \mathbf{v} \tag{35}$$

Several examples are provided here to demonstrate the capability of the DG-CVS solver to solve the level-set equation. In all the examples, the initial signed distance field is narrowed by the following

$$\psi = \begin{cases} -\epsilon & \text{if } \psi < -\epsilon \\ \psi & \text{if } -\epsilon \leq \psi \leq \epsilon \\ \epsilon & \text{if } \psi > \epsilon \end{cases}$$

whereϵ is chosen to be 0.1. This strategy is based on the assumption that the interface evolution often occurs within a narrow band.

7.1 Rotation of an off-center circular fluid body

The first example is about the rotation of a circular fluid body. The circular fluid body of radius 0.15 is centered at the location (0.5, 0.75). The circle is rotated by a velocity field $\mathbf{v} = (u, v)$ given by:

$$u = 2\pi(y - 0.5), \qquad \text{and } v = -2\pi(x - 0.5). \tag{36}$$

Such a velocity field completes one revolution per unit time. Figure 19 is the initial signed distance field where the zero level set is also indicated. The simulation is done on a 40×40 mesh. Figures 20 and 21 compares the $p1$ and $p3$ solutions at various times. Again, $p3$ solution preserves the circular shape better than the $p1$ solution.

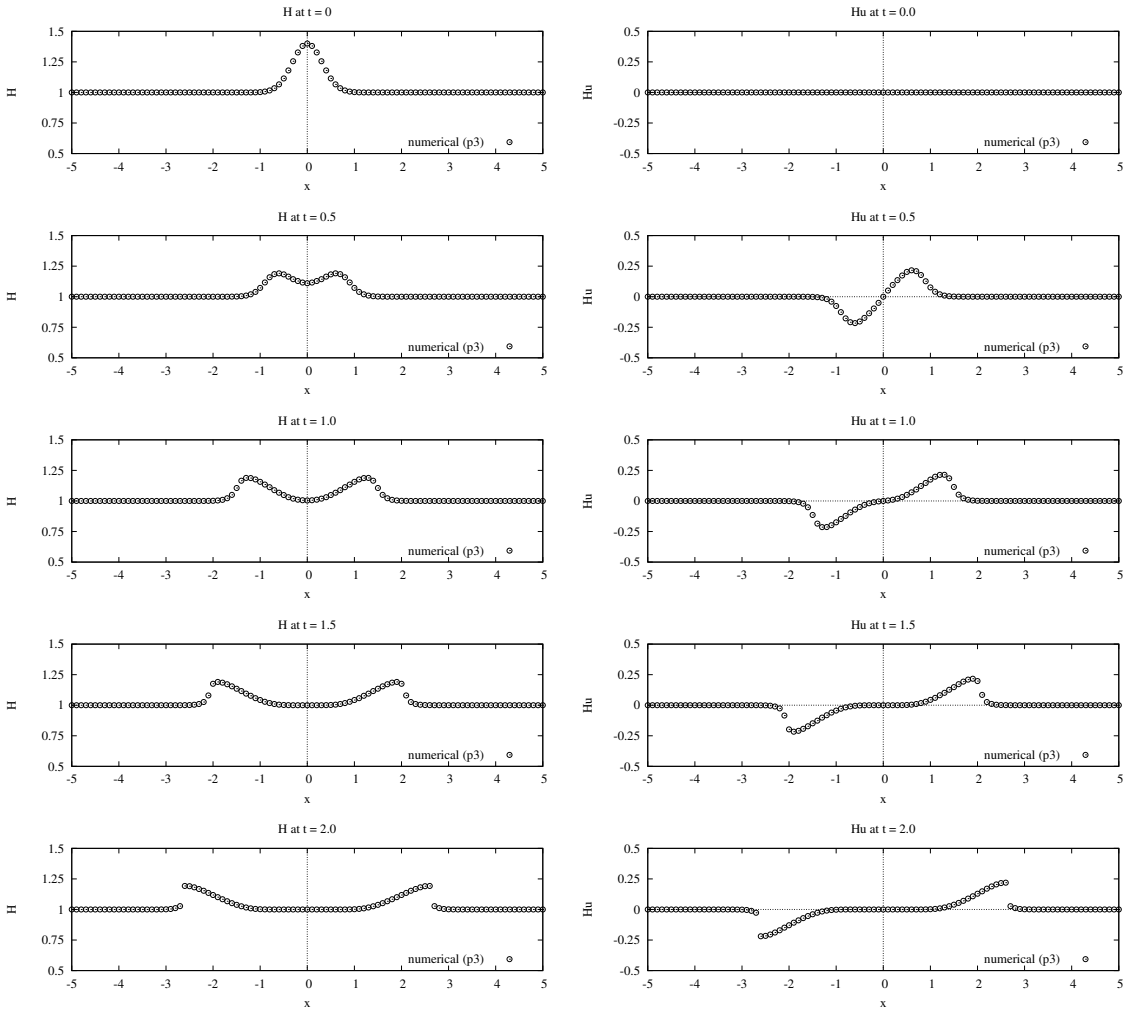


Figure 16: Line plots of the evolution of Gaussian pulse depth perturbation.

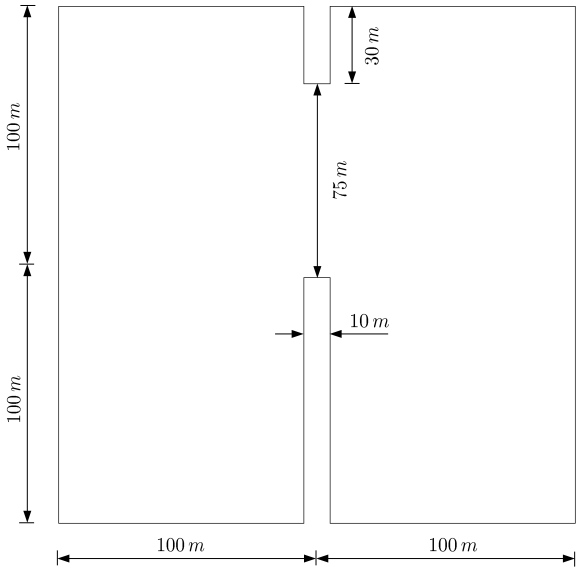


Figure 17: Geometry of the 2-D dam break problem.

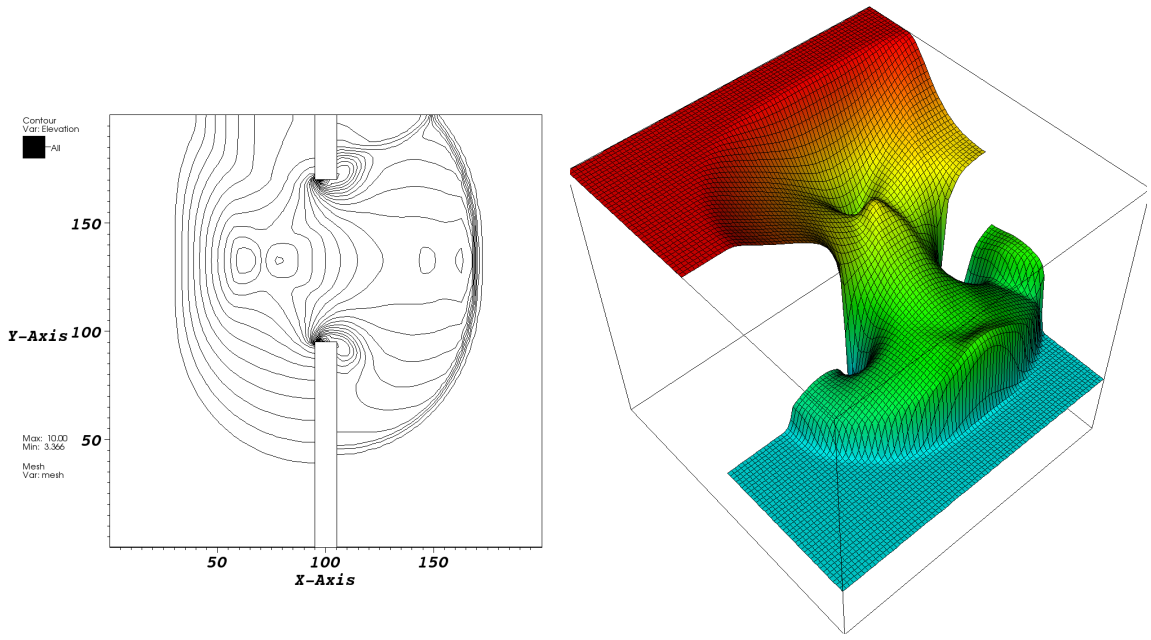


Figure 18: Contour lines and the carpet view of the water elevation of the 2-D dam break problem at $t = 7.2s$.

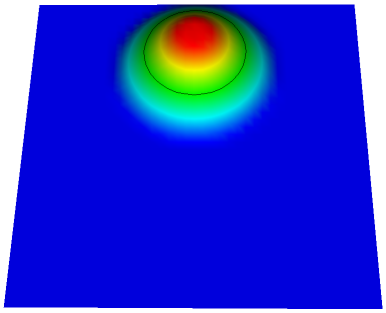
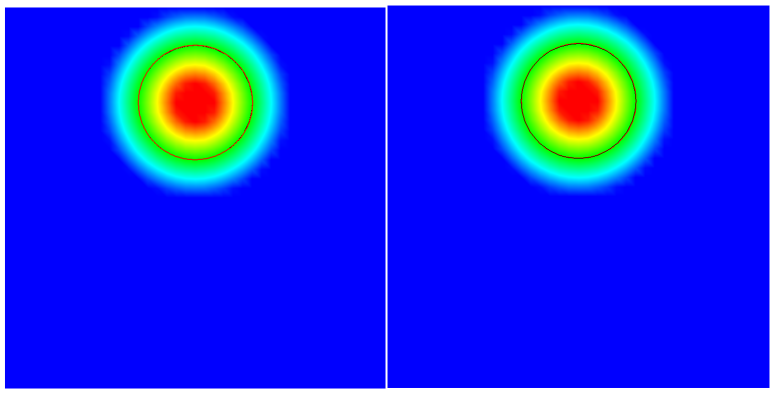
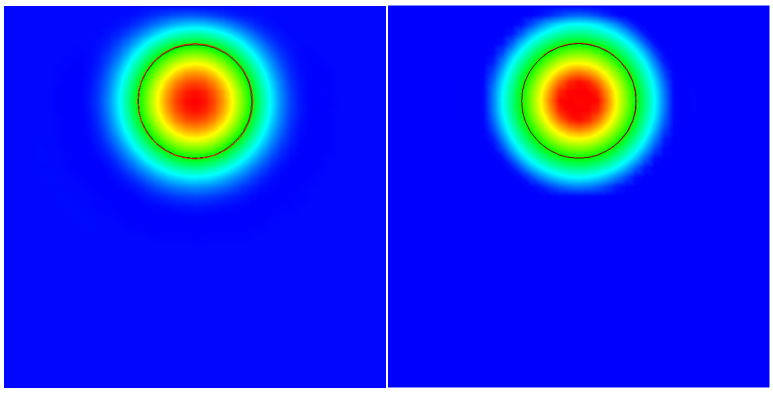


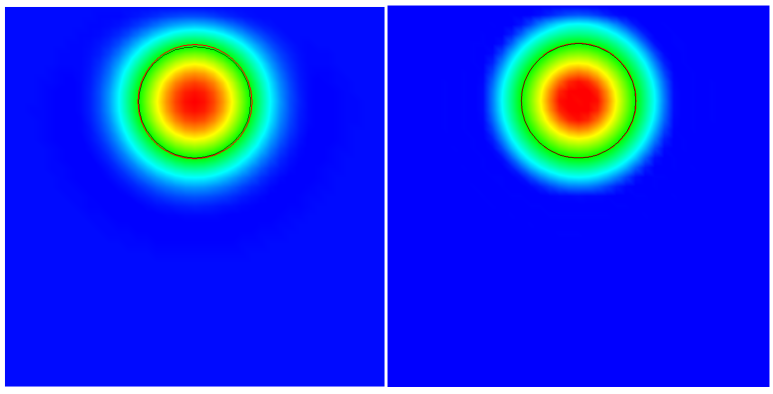
Figure 19: Initial signed distance field of an off-center circular fluid body.



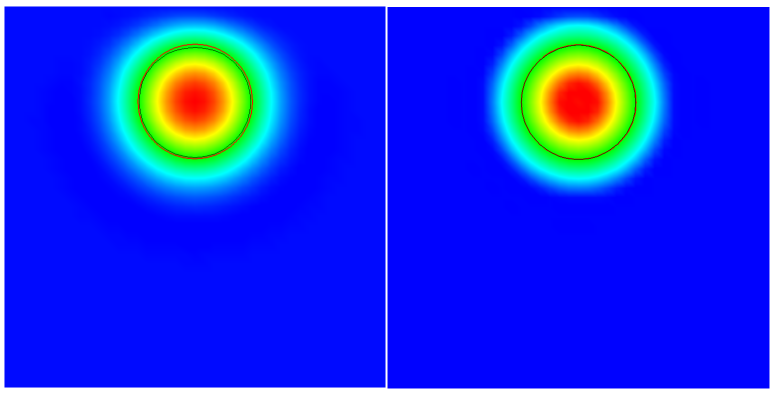
(a) $t = 0.0$.



(b) $t = 1.0$.

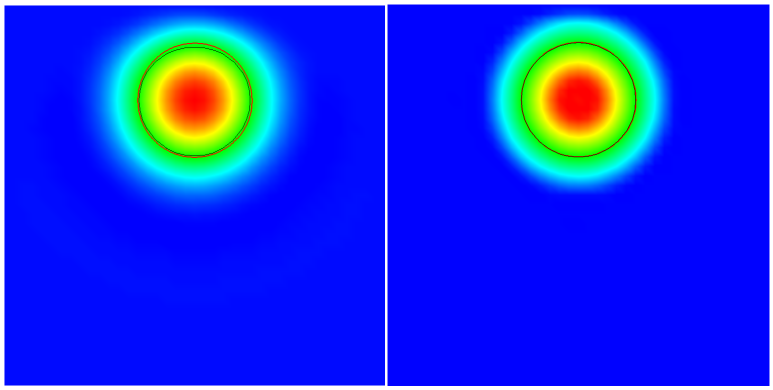


(c) $t = 2.0$.

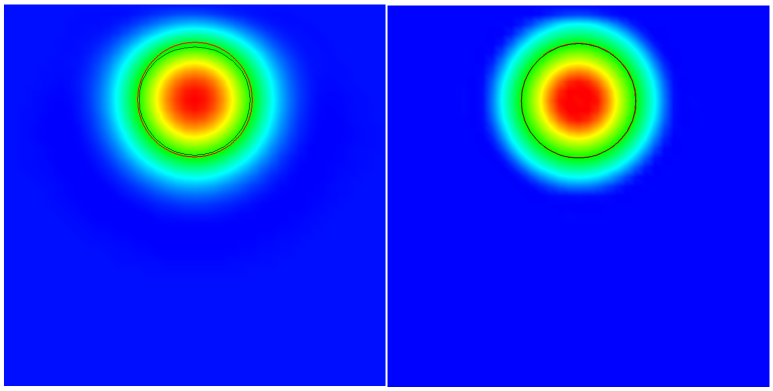


(d) $t = 3.0$.

Figure 20: Rotation of an off-center circular fluid body on a 40×40 rectangular mesh. Left column: $p1$ solutions and right column: $p3$ solutions. (to be continued)



(a) $t = 4.0$.



(b) $t = 5.0$.

Figure 21: Rotation of an off-center circular fluid body on a 40×40 rectangular mesh. Left column: $p1$ solutions and right column: $p3$ solutions. (continued)

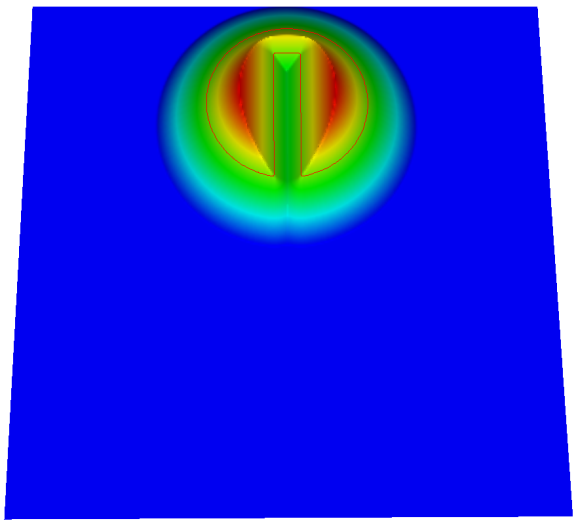


Figure 22: Zalesak disk. Initial level set is the signed distance to the disk.

7.2 Rotation of the slotted Zalesak disk

This another well known example is the rigid rotation of the Zalesak disk [19]. The initial interface is a circle centered at (0.5, 0.75) with a notch of width and depth 0.05 and 0.25, respectively. The velocity field is the same as the previous case, i.e. (36).

The initial level set function is initialized as the signed distance to the disk (Fig. 22). The zero level set indicates the interface location. Figure 23 shows the fourth order ($p3$) solution using the solver based on DG-CVS. The computational mesh contains 50×50 rectangular cells. The time step in the simulation is $\delta t = 0.00125$. 800 time steps were finish one revolution. Compared to the exact solution, the smearing is very small which indicates the small numerical dissipation of this high order scheme.

7.3 Shearing of a circular bubble

The last example is to simulate the deformation of a circular bubble of radius 0.15 centered at (0.5, 0.75). The circle is deformed due to the following velocity field [20]:

$$u = -\sin(2\pi y)\sin^2(\pi x)\cos(\pi t/T), \qquad \text{and } v = \sin(2\pi x)\sin^2(\pi y)\cos(\pi t/T) \tag{37}$$

which are obtained from $(-\partial\Psi/\partial y, \partial\Psi/\partial x)$ where Ψ is the following stream function

$$\Psi = \frac{1}{\pi}\sin^2(\pi x)\sin^2(\pi y).$$

At $t = T/2$, the velocity starts to reverse direction and at time $t = T$, the original circle is expected to be restored.

The velocity field given by (37) represents a vortex which stretch the circular fluid body into a thin filament toward the vortex center. The simulation is performed on a relatively coarse 40×40 mesh. Here $T = 4$. Figures 24 and 25 shows the fourth order ($p3$) evolution of the signed distance field and the zero level set at various times. At $t = 0.5T = 4$, the bubble starts to restore its shape. At $t = 4.0$, the bubble recovers its original circular shape. The comparison with the exact solution shows the accuracy of the current method.

8 Major Progress #5: Solving Moving Mesh Problems

The ultimate goal of developing DG-CVS based solvers is to solve fluid flow problems involving moving boundaries. Due to the space-time formulation of DG-CVS, DG-CVS automatically satisfies the so-called

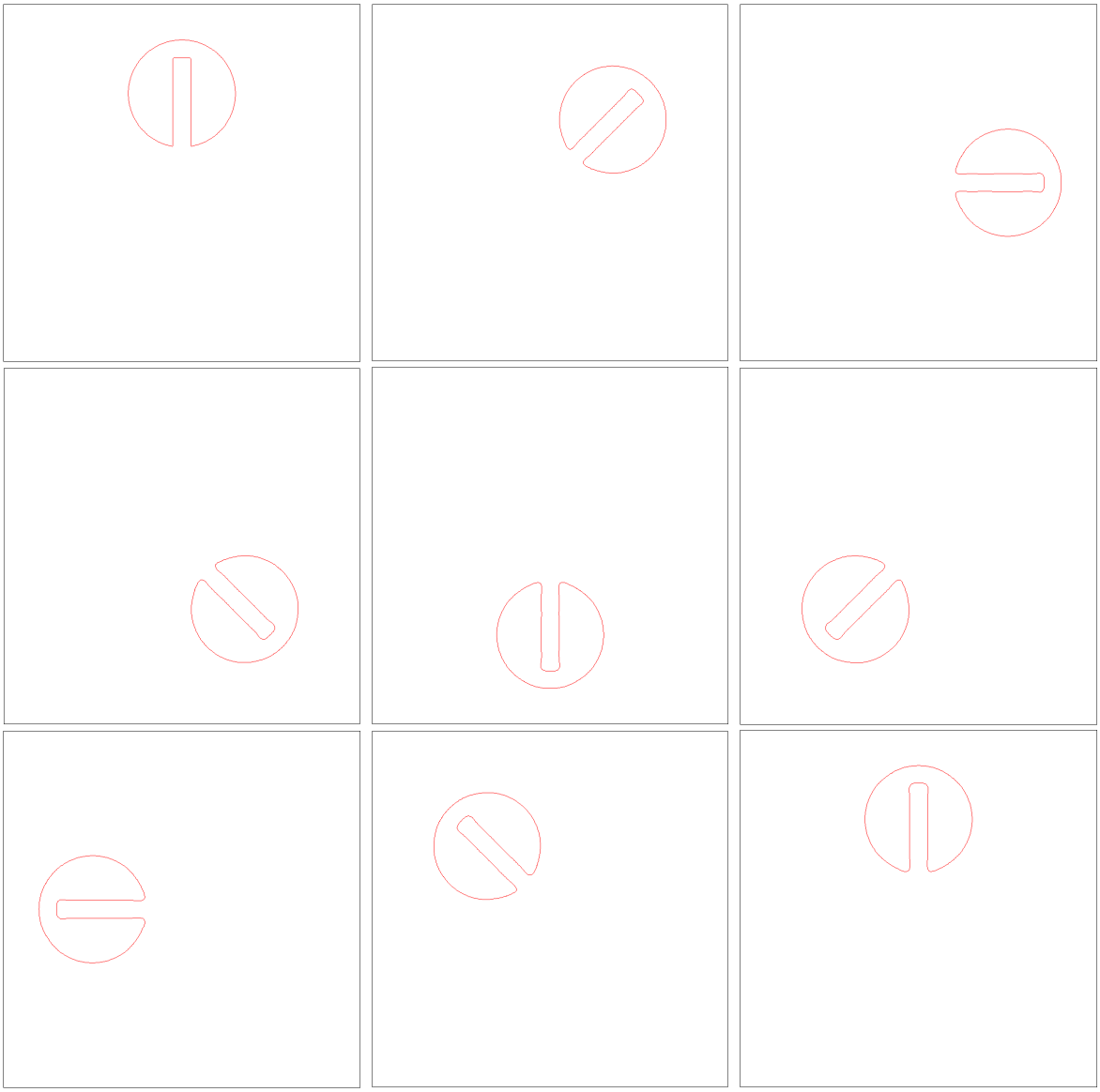
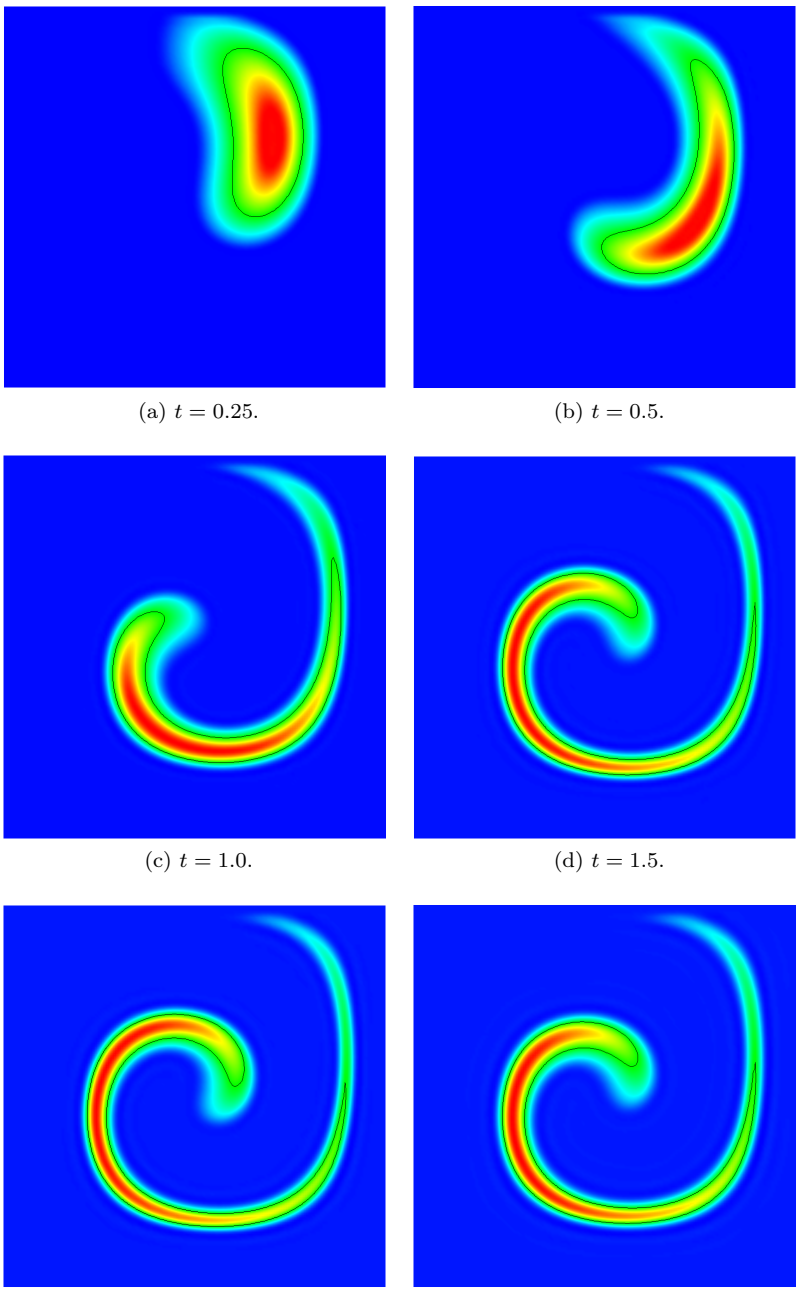


Figure 23: Rotation of Zalesak disk. The locations of the disk at several instants within one revolution.



(a) $t = 0.25$.

(b) $t = 0.5$.

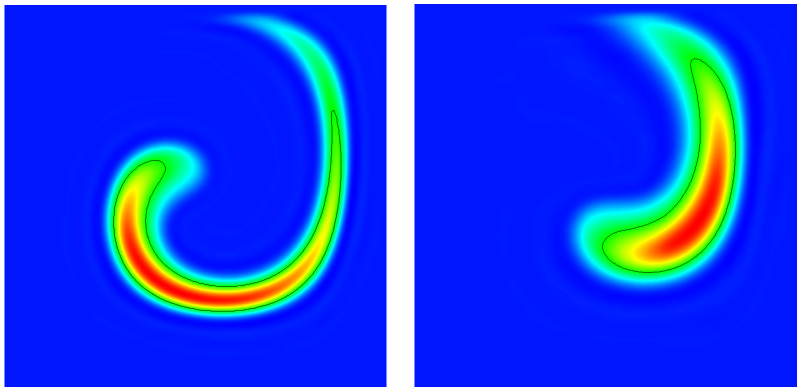
(c) $t = 1.0$.

(d) $t = 1.5$.

(e) $t = 2.0$.

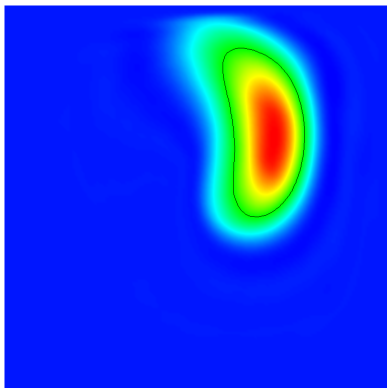
(f) $t = 2.5$.

Figure 24: $p3$ solution of the shearing of a circular bubble by a vortex on a 40×40 mesh. $T = 4$.

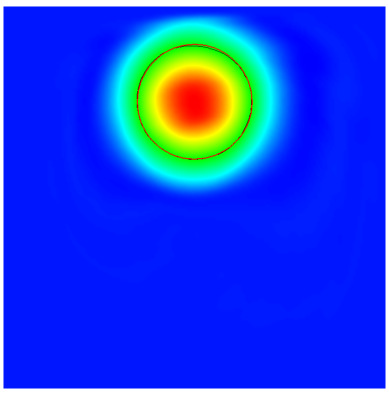


(a) $t = 3.0$.

(b) $t = 3.5$.



(c) $t = 3.75$.



(d) $t = 4.0$.

Figure 25: $p3$ solution of the shearing of a circular bubble by a vortex on a 40×40 mesh. $T = 4$.(continued)

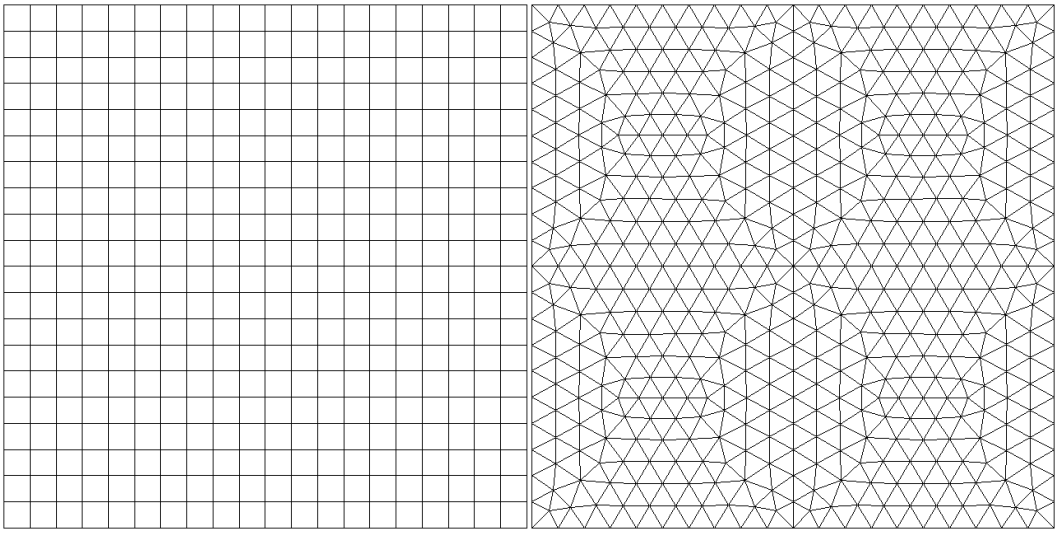


Figure 26: Meshes used in the simulation. Left: 20×20 rectangular mesh `qua-20`. Right: unstructured triangular mesh `tri-20` with similar edge resolution.

geometric conservation law (GCL) by incorporating the mesh motion effect into the unit normal of the space-time side surfaces of the conservation element during the enforcement of space-time fluxes. Therefore, any extra sophisticated techniques used to enforce the GCL in other semi-discrete high-order methods can be avoided.

A simple test is presented here to demonstrate the moving-mesh capability of the current DG-CVS solver. This test is an isentropic vortex advection problem on a 2D square domain $[0, 10] \times [0, 10]$. The initial conditions are given as an isentropic vortex with the center at $(5, 5)$, i.e.,

$$u(x, y, 0) = 1 - \frac{\epsilon}{2\pi} e^{0.5(1-r^2)}(y - 5) \tag{38a}$$

$$v(x, y, 0) = 1 + \frac{\epsilon}{2\pi} e^{0.5(1-r^2)}(x - 5) \tag{38b}$$

$$T(x, y, 0) = 1 - \frac{(\gamma - 1)\epsilon^2}{8\gamma\pi^2} e^{1-r^2} \tag{38c}$$

$$S(x, y, 0) = 1 \tag{38d}$$

where u, v, T, and S are x-velocity, y-velocity, temperature and entropy, respectively. $\epsilon = 5$ represents the vortex strength and $r^2 = (x - 5)^2 + (y - 5)^2$. The periodic boundary conditions on both directions are assumed. The density ρ and the pressure P can be obtained via

$$\rho(x, y, 0) = \left(\frac{T(x, y, 0)}{S(x, y, 0)} \right)^{1/(\gamma-1)}, \quad P(x, y, 0) = \rho(x, y, 0) T(x, y, 0)$$

where $\gamma = 1.4$ is the ratio of specific heats. It can be shown that the Euler equations with the above initial and boundary conditions allows an exact solution which is the initial solution advected with the speed $(1, 1)$ in the diagonal direction. With periodic boundary conditions, the vortex will come back to the center of the domain, its original location, at $t = 10$.

Both rectangular and triangular meshes are tested. Since we are going to present the fourth order ($p3$) solutions in this paper, relatively coarse meshes are used. The meshes are designated as `rec-20` and `tri-20` (cf. Fig. 26) where each boundary of the domain contains equally spaced 20 edges. The interior elements of the triangular meshes are unstructured and generated by EasyMesh [21].

In the simulation, the grid points on the boundaries are fixed and the interior grid points move according

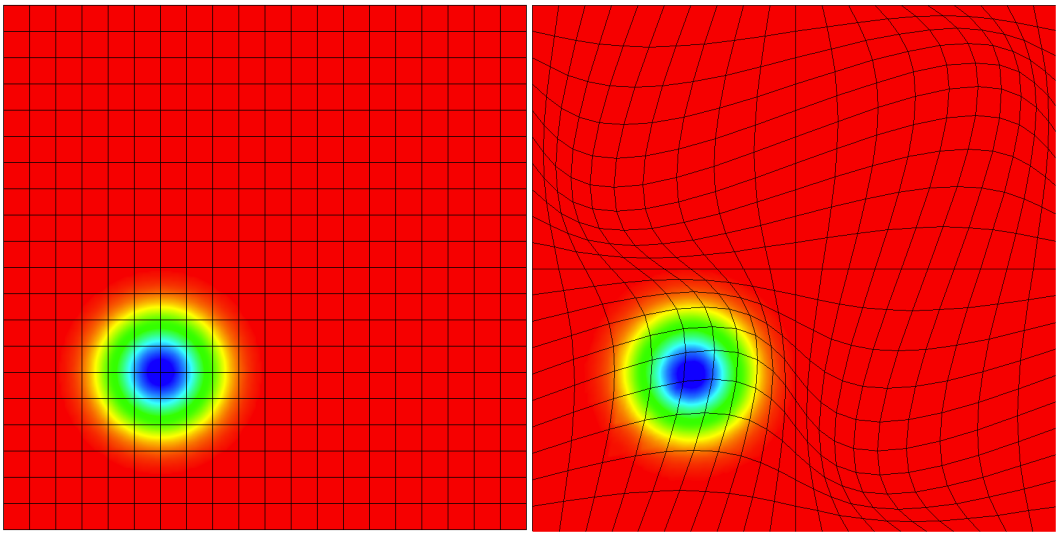


Figure 27: Comparison of the $p3$ density field at $t = 8$ of the vortex advection problem on mesh `qua-20`. Left: stationary mesh. Right: moving mesh.

to

$$x(\xi, \eta, t) = \xi + \sin(0.2\pi\xi)\sin(0.2\pi\eta)\sin(-0.2\pi t) \tag{39a}$$
$$y(\xi, \eta, t) = \eta + \sin(0.2\pi\xi)\sin(0.2\pi\eta)\sin(-0.2\pi t) \tag{39b}$$

where (ξ, η) is the spatial coordinate in the fixed reference domain which is the same as the physical domain at $t = 0$. The above mesh motion equations also ensure that the grid points on the vertical and horizontal center lines ($\xi = 5$ and $\eta = 5$, respectively) are not moving with time.

For fair comparison, the time step used in all simulations is taken as $\delta t = 0.02$, regardless of whether the mesh is rectangular or triangular and whether the mesh is stationary or moving. Only the fourth order ($p3$) results will be provided.

Figure 27 shows the comparison of the density fields at $t = 8$ on the stationary and the moving quadrilateral meshes. As can be seen, the solution on the moving mesh is visually as smooth as the one on stationary mesh. Figure 28 shows similar density fields on triangular meshes.

Figures 29 and 30 compare the density distribution at $t = 10$ along the horizontal center line ($y = 5$) of the domain with the exact solution for all simulations. The accuracy is clearly demonstrated.

Finally, we present the quantitative error comparison between all simulations in Table 14. Here, l_1, l_2 and l_∞ errors are evaluated at the discrete vertices of the mesh. The L_2 error is the integrated error over the top surfaces of the vertex-level CE. Recall that all simulations use the same time step $\delta t = 0.02$. As can be seen, for the same `qua-20` or `tri-20` meshes, the errors on moving meshes are larger than those on stationary meshes. The is expected because the current prescribed mesh motion is independent from the solution. Mesh motion causes the loss of grid resolution compared with the uniform stationary mesh (cf. Figs. 27 and 28). In addition, the errors on triangular meshes are smaller than those on rectangular meshes. This is because there are more cells on the triangular mesh even if its edge resolution is comparable to that of the rectangular mesh. Therefore, triangular meshes yield more accurate results at the price of higher computational cost.

9 Acknowledgment/Disclaimer

This work was sponsored by the Air Force Office of Scientific Research under grant number FA9550- 10-1-0045. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air

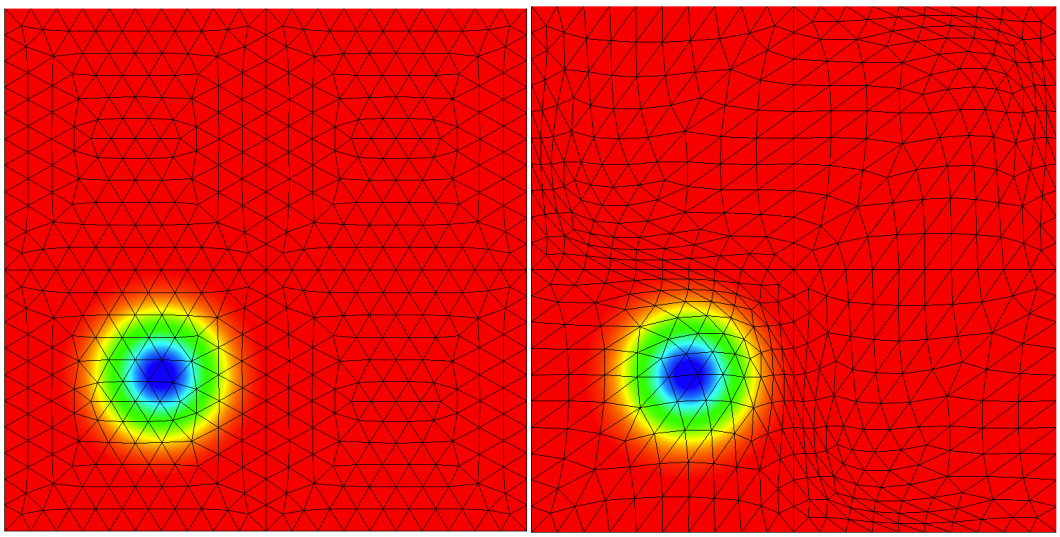


Figure 28: Comparison of the $p3$ density field at $t = 8$ of the vortex advection problem on `mesh tri-20`. Left: stationary mesh. Right: moving mesh.

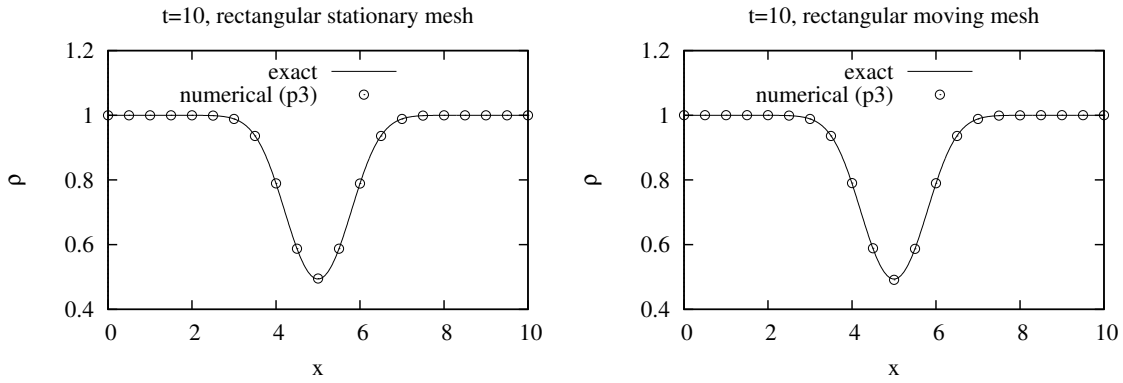


Figure 29: Comparison of the $p3$ density distribution of the vortex advection problem at $t = 10$ along the horizontal center line of the domain on mesh `qua-20`. Left: stationary mesh. Right: moving mesh.

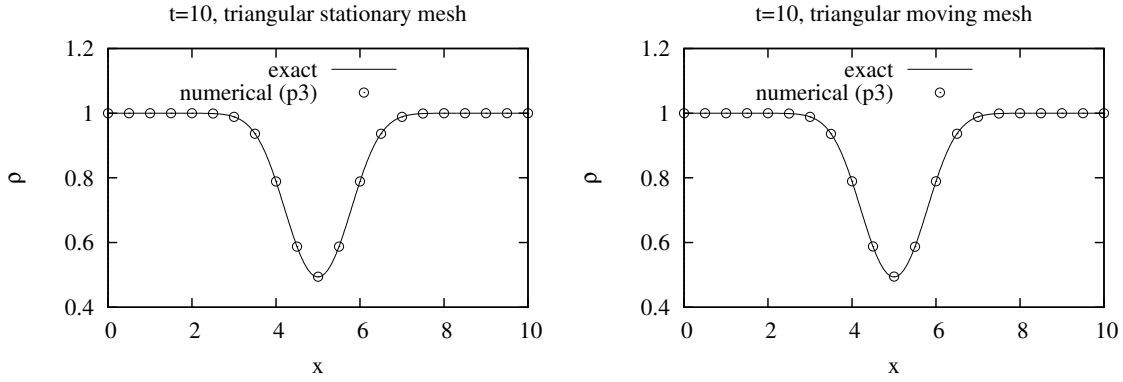


Figure 30: Comparison of the $p3$ density distribution of the vortex advection problem at $t = 10$ along the horizontal center line of the domain on mesh `tri-20`. Left: stationary mesh. Right: moving mesh.

Table 14: Comparison of the $p3$ density errors of the isentropic vortex advection problem on fixed and moving meshes.

	fixed qua-20 mesh	moving qua-20 mesh	fixed tri-20 mesh	moving tri-20 mesh
l_1-error	4.32e-05	2.54e-04	2.13e-05	8.41e-05
l_2-error	1.19e-04	4.22e-04	3.77e-05	1.80e-04
l_∞-error	1.50e-03	3.36e-03	4.82e-04	1.68e-03
L_2-error	1.12e-04	4.58e-04	4.15e-05	1.84e-04

Force Office of Scientific Research or the U.S. Government. The authors are also grateful to the School of Engineering and the Department of Computer Engineering at Jackson State University for their support.

10 Publications

This project has produced the following full-length journal and conference articles.

1. **S. Tu**, G.W. Skelton and Q. Pang, "Extension of the High-Order Space-Time discontinuous Galerkin Cell Vertex Scheme for Time Dependent Diffusion Equations," *Communications in Computational Physics*. Vol. 11, No. 5, (2012) pp 1503-1524.
2. **S. Tu** and Q. Pang, "Development of the High-Order Space-Time Discontinuous Galerkin Cell Vertex Scheme (DG-CVS) for Moving Mesh Problems," presented at the 2012 AIAA Fluid Dynamics Conference in New Orleans, LA. AIAA Paper 2012-2835.
3. **S. Tu**, Q. Pang and H. Xiang, "Solving the Level Set Equation Using the High Order Space-Time Discontinuous Galerkin Cell-Vertex Scheme," presented at the 50th Aerospace Science Meetings, Jan. 9-12, Nashville, TN, AIAA Paper 2012-1233.
4. **S. Tu**, Q. Pang and H. Xiang, "Solving the Shallow Water Equations Using the High Order Space-Time Discontinuous Galerkin Cell-Vertex Scheme," presented at the 50th Aerospace Science Meetings, Jan. 9-12, Nashville, TN, AIAA Paper 2012-0307.
5. S**. Tu**, Q. Pang and H. Xiang, "Wave Computation Using a A High Order Space-time Riemann Solver Free Method," presented at the 17th AIAA/CEAS Aeroacoustics Conference, Portland, Oregon, June, 2011. AIAA Paper 2011-2846.
6. **S. Tu**, G.W. Skelton, and Q. Pang, "Solving Time Dependent Diffusion Equations on Arbitrary Grids Using the High-Order Space-Time discontinuous Galerkin Cell Vertex Scheme," AIAA Paper 2011-0050. Presented at the 49th AIAA Aerospace Science Meeting, Jan. 4-7, 2011, Orlando, FL.
7. **S. Tu**, G. Skelton and Q. Pang, "A Compact High-order Space-time Method for Conservation Laws," *Communications in Computational Physics*, Vol. 9, No. 2, (2011) pp. 441-480.
8. **S. Tu** and Z. Tian, "Preliminary Implementation of a High Order Space-time Method on Overset Cartesian/Quadrilateral Grids," presented at the 48th AIAA Aerospace Science Meeting, Jan. 4-7, 2010, Orlando, FL. AIAA Paper 2010-0544.
9. **S. Tu**, "A Solution Limiting Procedure for an Arbitrarily High Order Space-Time Method," presented at the 19th AIAA Computational Fluid Dynamics Conference, Jun. 22-25, 2009, San Antonio, TX, AIAA Paper 2009-3983.
10. **S. Tu**, "A High-order Space-time Method for Compressible Euler Equations," presented at the 47th AIAA Aerospace Science Meeting, Jan. 5-8, 2009, Orlando, FL. AIAA Paper 2009-1335.

11 Personnel Supported

- Shuangzhang Tu, faculty member (PI).
- Haibin Xiang, Master's graduate student.
- Yongze Liu, Master's graduate student.

12 AFOSR Point of Contact

Dr. Fariba Fahroo, Program Manager, Computational Mathematics, AFOSR/NL, 875 North Randolph Street Suite 325, Room 3112 Arlington, VA 22203, (703) 696-8429, Fax (703) 696-8450, DSN 426-8429, fariba.fahroo@afosr.af.mil.

13 Interactions/Transitions

Conference presentations since April 2010 when the project started:

1. One talk at the AIAA Fluid Dynamics Conference in New Orleans, LA, June 2012. (speaker: S. Tu).
2. Two talks at the 50th Aerospace Science Meetings, Jan. 9-12, 2012, Nashville, TN. (speaker: S. Tu).
3. One talk at the first High Order CFD Methods Workshop, Jan. 7-8, 2012, Nashville, TN. (speaker: S. Tu).
4. One talk at the 7th International Congress on Industrial and Applied Mathematics (ICIAM 2011), July 18-22, 2011, Vancouver, BC, Canada. (speaker: S. Tu).
5. One talk at the 17th AIAA/CEAS Aeroacoustics Conference, Portland, Oregon, June, 2011. (speaker: S. Tu).
6. One talk at the 49th AIAA Aerospace Science Meeting, January, 2011, Orlando, FL. (speaker: S. Tu).
7. One invited talk in the Department of Mathematics at the Air Force Institute of Technology, October 7, 2010, Dayton, OH. (speaker: S. Tu)
8. One invited talk at the Engineering Professional Development Seminar, IEEE Mississippi Section, August 20, 2010. (speaker: S. Tu)

14 Changes in Research Objectives

None.

15 Change in AFOSR Program Manager

None.

16 Extensions Granted or Milestones Slipped

None.

17 New Discoveries

None patentable.

References

[1] Chang, S.-C. and To, W., "A new numerical framework for solving conservation laws: the method of space-time conservation element and solution element," 1991, NASA TM 1991-104495.

[2] Cockburn, B. and Shu, C.-W., "Runge-Kutta discontinuous Galerkin methods for convection-dominated problems," *J. Sci. Comput.*, Vol. 16, No. 3, 2001, pp. 173–261.

[3] Cockburn, B., Hou, S., and Shu, C.-W., "TVB Runge-Kutta local projection discontinuous Galerkin method for conservation laws IV: the multidimensional case," *Math. Comp.*, Vol. 54, 1990, pp. 545–581.

[4] Dumbser, M., Käserb, M., Titarevb, V. A., and Toro, E. F., "Quadrature-free non-oscillatory finite volume schemes on unstructured meshes for nonlinear hyperbolic systems," *J. Comp. Phys.*, Vol. 226, No. 1, September 2007, pp. 204–243.

[5] Dyson, R., "Technique for Very High Order Nonlinear Simulation and Validation," 2001, NASA/TM—2001-210985.

[6] Dasgupta, G., "Integration within polygonal finite elements," *J. Aerosp. Engrg.*, Vol. 16, No. 1, 2003, pp. 9–18.

[7] Rathod, H. and Rao, H., "Integration of trivariate polynomials over linear polyhedra in Euclidean three-dimensional space," *J. Austral. Math. Soc. Ser.*, Vol. 39, 1998, pp. 355–385.

[8] Atkins, H. and Shu, C.-W., "Quadrature-free implementation of discontinuous Galerkin method for hyperbolic equations," *AIAA J.*, Vol. 36, No. 5, 1998, pp. 775–782.

[9] Harris, R., Wang, Z., and Liu, Y., "Efficient quadrature-free high-order spectral volume method on unstructured grids: Theory and 2D implementation," *J. Comp. Phys.*, Vol. 227, 2008, pp. 1620–1642.

[10] Tu, S., Skelton, G., and Pang, Q., "A compact high order space-time method for conservation laws," *Communications in Computational Physics*, Vol. 9, No. 2, 2011, pp. 441–480.

[11] Baumann, C. and Oden, J., "A discontinuous hp finite element method for convection-diffusion problems," *Comput. Methods Appl. Mech. Engrg.*, Vol. 175, 1999, pp. 311–341.

[12] Babuška, I., Baumann, C., and Oden, J., "A discontinuous hp finite element method for diffusion problems: 1-D analysis," *Computers and Mathematics with Applications*, Vol. 37, No. 9, 1999, pp. 103–122.

[13] Larson, M. G. and Niklasson, A. J., "Analysis of a family of discontinuous Galerkin methods for elliptic problems: the one dimensional case," *Numer. Math.*, Vol. 99, November 2004, pp. 113–130.

[14] Romkes, A., Prudhomme, S., and Oden, J. T., "Convergence analysis of a discontinuous finite element formulation based on second order derivatives," *Comput. Meth. Appl. Mech. Eng.*, Vol. 195, 2006, pp. 3461–3482.

[15] Dolejší, V. and Havle, O., "The L2-Optimality of the IIPG Method for Odd Degrees of Polynomial Approximation in 1D," *J. Sci. Comput.*, Vol. 42, 2010, pp. 122–143.

[16] Liu, Y., Shu, C.-W., Tadmor, E., and Zhang, M., "Central Local Discontinuous Galerkin Methods on Overlapping Cells for Diffusion Equations," 2010, submitted to Mathematical Modelling and Numerical Analysis and under revision.

[17] Eriksson, K., Estep, D., Hansbo, P., and Johnson, C., *Computational Differential Equations*, Cambridge University Press, 1996.

[18] Popescu, M., Shyy, W., and Garbey, M., "Finite volume treatment of dispersion-relation-preserving and optimized prefactored compact schemes for wave propagation," *J. Comput. Phys.*, Vol. 210, No. 2, 2005, pp. 705–729.

[19] Zalesak, S., "Fully multidimensional flux-corrected transport algorithms for fluids," *J. Comput. Phys.*, Vol. 31, 1979, pp. 335–362.

[20] Bell, J. B., Colella, P., and Glaz, H. M., "A second-order projection method for the incompressible Navier-Stokes equations," *J. Comput. Phys.*, Vol. 85, 1989, pp. 257–283.

[21] Niceno, B., "EasyMesh," http://www-dinma.univ.trieste.it/nirftc/research/easymesh/.